Les Cahiers du GERAD ISSN: 0711–2440

The fragility-constrained vehicle routing problem with time windows

C. Altman, G. Desaulniers, F. Errico

G–2020–78

December 2020

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

The fragility-constrained vehicle routing problem with time windows

Clément Altman [a]

Guy Desaulniers [a]

Fausto Errico [b]

[a] *GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

[b] *GERAD & CIRRELT & Departement of Civil Engineering, École de Technologie Supérieure de Montréal, Montréal (Québec) Canada, H3C 2E5*

clement.altman@polytechnique.org
guy.desaulniers@gerad.ca
fausto.errico@etsmtl.ca

December 2020
Les Cahiers du GERAD
G–2020–78

Abstract : We study a new variant of the well-studied Vehicle Routing Problem with Time Windows (VRPTW), called the fragility-constrained VRPTW, which assumes that 1) the capacity of a vehicle is organized in multiple identical stacks; 2) all items picked up at a customer are either "fragile" or not; 3) no non-fragile items can be put on top of a fragile item (the *fragility* constraint); and 4) no en-route load rearrangement is possible. We first characterize the feasibility of a route with respect to this fragility constraint. Then, to solve this new problem, we develop an exact branch-price-and-cut (BPC) algorithm that includes a labeling algorithm exploiting this feasibility characterization to efficiently generate feasible routes. This algorithm is benchmarked against another BPC algorithm that deals with the fragility constraint in the column generation master problem through infeasible path cuts. Our computational results show that the former BPC algorithm clearly outperforms the latter in terms of computational time and that the fragility constraint has a greater impact on the optimal solution cost (compared to that of the VRPTW) when vehicle capacity decreases, stack height increases and for a more balance mix of customers with fragile and non-fragile items.

Keywords: Vehicle routing, multiple stacks, fragility loading constraint, branch-price-and-cut, route feasibility characterization

1 Introduction

This paper introduces a new variant of the classical Vehicle Routing Problem with Time Windows (VRPTW, see Desaulniers et al. 2014) where a particular *fragility* loading constraint is enforced. This variant is called the Fragility-constrained VRPTW (F-VRPTW). The classical VRPTW considers a set of customers, each specifying a demand volume to be picked up and a pickup time window. Pickups are performed by a fleet of homogeneous vehicles with given capacity, initially located at a depot. Travel times and costs among customers and between each customer and the depot are assumed to be known. The VRPTW consists in computing least-cost vehicle routes such that each customer is visited exactly once within the indicated time window, while the vehicle capacity is not exceed for each route. It has been extensively studied in the literature (see Desaulniers et al. 2014) and the current state-of-the-art exact methodology for solving it is branch-price-and-cut (BPC, see Costa et al. 2019). In particular, the sophisticated BPC algorithms of Pecin et al. (2017a) and Sadykov et al. (2020) are able to solve to proven optimality most tested instances with up to 200 customers.

For the F-VRPTW, the vehicle capacity is organized in multiple identical stacks of a given maximal height. Customers might have items labeled as "fragile". To avoid potential damages to these items, the F-VRPTW imposes the fragility constraint, which implies the following: 1) non-fragile items are prevented to be stacked on top of fragile ones, however 2) fragile items are allowed to be stacked on top of other fragile items. Furthermore, no en-route load rearrangement is allowed. This problem is cast as a pickup one but the developed methodology can easily be adapted for a delivery problem, simply by inverting the role of the fragile and non-fragile items. Our motivation in studying the F-VRPTW is its relevance for naval shipping companies delivering freight in northern Quebec and, to the best of our knowledge, this problem has never been studied before. However, although in a different loading framework (three-dimensional loading), the fragility constraint, in the exact sense used in the present work, was already introduced by Gendreau et al. (2006). Furthermore, the F-VRPTW is related to a number of studies addressing routing problems combined with several types of loading constraints. For an extensive literature review, the reader is referred to Pollaris et al. (2015) and references therein.

A first stream of work related to the F-VRPTW focuses on routing problems where the vehicle capacity is organized in multiple stacks (see Côté et al. 2012; Carrabs et al. 2013; Cherkesly et al. 2016; Veenstra et al. 2017, to cite a few). Handling multiple stacks responds to practical concerns in a variety of contexts, such as grocery, pharmaceutical and naval applications where either multi-compartments vehicles are required, or freight need to be organized into pallets or containers which are then stacked for shipping. From the operational viewpoint, structuring the load in stacks limits the access to the only items located on the top of the stacks, unless time-consuming rearrangements of the freight are operated. In real-world operations, this limitation results in the so-called LIFO policy. Typically, the LIFO policy is a concern when the underlying routing decisions belong to the family of the Pickup and Delivery Problems (PDPs). In fact, the pickup order will influence which items must be delivered first, thus deeply impacting routing possibilities. In the VRPTW literature, however, the LIFO policy is generally not a concern because all the freight either originates from the depot (in case of deliveries) where it can be well positioned in the vehicle before departure or is destined to the depot (in case of pickups) where it can be typically unloaded in any order. Due to the structural differences between the PDPs and the VRPTW, providing an extensive literature review on the PDPs is out of the scope, and we again refer the interested reader to Pollaris et al. (2015). However, we notice that, given the additional fragility constraint in the F-VRPTW, the LIFO rule may render infeasible some customer sequences, and this needs to be taken into account when solving the F-VRPTW.

Another related stream of literature addresses the *load-bearing strength* constraint, which can be seen, to some extent, as a generalization of the fragility constraint introduced here. This constraint assures that the total weight stacked on top of a given item does not exceed a maximal value, thus preventing item damages. Junqueira et al. (2013) proposes an integer linear programming model for the vehicle routing problem with three-dimensional loading constraints (3L-CVRP), including the load-bearing strength. By adopting a commercial mixed-integer programming solver, the authors obtained the solution for some small problem instances. More recently, a single-stack variant of this problem was addressed by Chabot et al. (2017) in the context of warehouse order picking activities. The authors propose several heuristics as well as exact branch-and-cut algorithms. As previously mentioned, Gendreau et al. (2006) address the 3L-CVRP where the fragility constraint is handled as in our setting. For this problem, the authors developed an efficient Tabu Search (TS) algorithm. Different metaheuristics have been developed for the same problem by several other authors, see Fuellerer et al. (2010); Tarantilis et al. (2009); Tao and Wang (2015), to cite a few.

In this paper, we propose a set partitioning formulation of the F-VRPTW, which we solve via a BPC algorithm. The most challenging aspect is in the solution of the column generation pricing problem, which turns out to be a variant of the Elementary Shortest Path Problem with Resource Constraints (ESPPRC). The main contribution of this paper is in the way we handle the fragility constraint. We solve the pricing problem by a labeling algorithm and provide a formal characterization of partial routes allowing us to easily check their feasibility with respect to the fragility constraint. In particular, this characterization allows us to avoid the intuitive but naive approach of duplicating labels to account for alternative freight configurations. To improve the performance of the labeling algorithm, we develop an efficient dominance rule with the purpose of discarding proven dominated labels.

To analyze the performance of our algorithm, we perform an extensive computational campaign. We first introduce F-VRPTW instances that are obtained from the Solomon's VRPTW benchmark instances by varying the vehicle capacity and the maximal stack height. Our computational results show that the fragility constraint is more binding when vehicle capacity is smaller (i.e., for shorter routes) and when stacks are higher. Furthermore, to assess the performance of the proposed BPC algorithm, we have developed an alternative solution method, which is a classic BPC algorithm for the VRPTW (possibly generating routes violating the fragility constraint), augmented with infeasible path cuts to eliminate integer solutions violating the fragility constraint. Our test results show that the proposed BPC algorithm consistently outperforms the alternative one.

The rest of the paper is organized as follows. In Section 2, we formally state the F-VRPTW and present a mathematical formulation. In Section 3, we prove the main theoretical result of the paper, that gives a formal characterization of a route feasible with respect to the fragility constraint. This result is then exploited in Section 4 to develop an efficient BPC algorithm. We report and analyze computational results in Section 5, before providing concluding remarks in Section 6.

2 Problem statement and mathematical formulation

The classical VRPTW can be described as follows. Consider a directed graph $G = (\mathcal{N}, \mathcal{A})$ where $\mathcal{N} = \{0, 1, \ldots, n, n+1\}$ is the node set and $\mathcal{A} = \{(i,j) \mid i,j \in N,\ i \neq j, (i,j) \neq (n+1, 0)\}$ is the arc set. We denote $\mathcal{N}_c = \{1, \ldots, n\}$ the subset of nodes representing the n customers, while nodes 0 and $n+1$ represent the same depot at the beginning and the end of a route, respectively. A fleet of homogeneous vehicles with capacity Q is initially located at the depot. A demand volume q_i (an integer number of items of the same size) to be collected and a time window $[\underline{w}_i, \overline{w}_i]$ are associated with each customer $i \in \mathcal{N}_c$. A vehicle is allowed to arrive at customer i earlier than $\underline{w}_i$, but must leave in the specified time window. A cost c_{ij} and a travel time t_{ij} (possibly including a service time at i) are associated with each arc $(i,j) \in \mathcal{A}$. We assume that these costs and travel times satisfy the triangle inequality. Observe that set $\mathcal{A}$ can be reduced by eliminating all arcs (i,j) with $i,j \in \mathcal{N}_c$ that are load or time infeasible, i.e., such that $q_i + q_j > Q$ or $\underline{w}_i + t_{ij} > \overline{w}_j$. The VRPTW calls for finding a minimum-cost set of routes starting and ending at the depot, such that each customer is visited exactly once, and the time windows and vehicle capacity constraints are fulfilled.

The F-VRPTW differs from the VRPTW in that the vehicle capacity is organized in a set of m identical stacks $M = \{1, \ldots, m\}$, each with a set of positions $K = \{1, \ldots, k\}$ (see Figure 1). Consequently, k is the height of a stack and the vehicle capacity is set as $Q = km$. Items are assumed to be put in the vehicle from the top and each of them occupies one position in one stack. Furthermore, the customer set $\mathcal{N}_c$ is partitioned in two subsets $\mathcal{L}$ and $\mathcal{H}$, identifying customers with fragile (light) and non-fragile (heavy) items, respectively. The fragility constraint requires that no item from customers

height $k = 3$	L	L		
2	H	H	L	
1	H	L	L	H
	1	2	3	4

$m = 4$ stacks

Figure 1: An infeasible loading configuration for a vehicle with 4 stacks of height 3

in $\mathcal{H}$ can be stacked on the top of items from customers in $\mathcal{L}$. However, fragile items can be stacked on top of other fragile ones. For example, the positioning of the items in stack 2 in Figure 1 is infeasible, while the positioning in the other stacks is legitimate. No en-route rearrangement of items is permitted. It should be noticed that the case of customers having both fragile and non-fragile items can be easily reduced to the present setting by adding suitable dummy nodes.

We formulate the F-VRPTW as a set-partitioning problem. To this scope, consider a route r as a sequence of nodes $r = (v_0, v_1, \ldots, v_p, v_{p+1})$, where $v_0 = 0$ and $v_{p+1} = n + 1$ represent the depot, and let Ω be set of all feasible routes. For all $i \in \mathcal{N}_c$, let a_{ir} be a parameter with value 1 if route r visits customer i and 0 otherwise. The cost c_r of route $r \in \Omega$ is computed as

$$c_r = \sum_{i=0}^{p} c_{v_i, v_{i+1}}. \tag{1}$$

By introducing a binary variable x_r for each route $r \in \Omega$ that takes value 1 if route r is chosen and 0 otherwise, the F-VRPTW can be formulated as follows:

$$\min \quad \sum_{r \in \Omega} c_r x_r \tag{2}$$

$$s.t. \quad \sum_{r \in \Omega} a_{ir} x_r = 1, \qquad \forall i \in \mathcal{N}_c \tag{3}$$

$$x_r \in \{0, 1\}, \qquad \forall r \in \Omega, \tag{4}$$

where the objective function (2) minimizes the total cost, constraints (3) ensure that each customer is visited exactly once, and constraints (4) impose that the variables are binary.

3 Characterization of fragility-feasible routes

The present section is devoted to introducing a formal characterization of a route that is feasible with respect to the fragility constraint (hereafter called a fragility-feasible route). As previously mentioned, the proposed solution method heavily stands on this characterization. In particular, as it will be clear in Section 4, this characterization allows us to drastically reduce the number of labels generated in the labeling algorithm developed to solve the column generation pricing problem.

The main observation behind the proposed characterization is that a route can be infeasible with respect to the fragility constraint only if, in the collecting process, we must form an incomplete stack containing fragile items, and then we keep collecting a large enough number of non-fragile items so that some of them must be placed on the incomplete stack with fragile items, hence violating the fragility constraint. As a consequence, the number of non-fragile items that can be collected after a given customer is bounded by a function depending on the number of fragile and non-fragile items collected so far and the order in which they were collected. However, when it is possible to complete all the stacks containing fragile items at a given customer with fragile items, the number of non-fragile items that can still be collected is only bounded by the vehicle capacity constraint.

Given a route $r = (v_0, v_1, \ldots, v_p, v_{p+1})$, let us consider the induced sequence of individual collected items $S_r = (b_1, b_2, \ldots, b_s)$. Given that customers belong either to set $\mathcal{L}$ or $\mathcal{H}$, the order used to embed the items of a given customer in sequence S_r is irrelevant. We denote by $a^{\mathcal{L}}(i)$ the number of fragile items in S_r until and including item b_i, by $a^{\mathcal{H}}(i)$ the corresponding number of non-fragile items, and by $l(i) = a^{\mathcal{L}}(i) + a^{\mathcal{H}}(i)$ the total number of collected items. Even if $l(i) = i$ here, notation $l(i)$ is used because the items will not always be considered individually in the rest of this paper. Let us also define a modified modulo function as:

$$F(x) = \begin{cases} x \bmod k & \text{if } x \bmod k \neq 0, \\ k & \text{otherwise.} \end{cases} \tag{5}$$

Finally, by a slight abuse of notation, we will write $b_i \in \mathcal{L}$ or $b_i \in \mathcal{H}$ to express that b_i is a fragile or non-fragile item, respectively.

Proposition 1 (Characterization) *Consider a route r and let us assume that r is feasible with respect to the capacity constraint. The corresponding sequence $S_r = (b_1, b_2, \ldots, b_s)$ is feasible with respect to the fragility constraint if and only if, for all $i \in \{1, \ldots, s\}$ with $b_i \in \mathcal{L}$, either*

(c1) $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) \geq k$

or

(c2) $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) \leq U(i)$,

where $a^{\mathcal{H}}(s)$ denotes the total number of non-fragile items and $U(i) = mk - (l(i) - l(i) \bmod k + k)$ is an upper bound on the number of non-fragile items that can be collected after item b_i.

Before proving this proposition, let us present the intuition behind conditions $(c1)$ and $(c2)$, and how they can be interpreted. Each fragile item $b_i \in \mathcal{L}$ in S_r may induce an upper bound on the number of non-fragile items that can be collected after it. This upper bound corresponds to the optimal value of the problem of finding a feasible loading configuration for the first i items in S_r that maximizes the number of positions available to load additional non-fragile items. Note that this upper bound does not depend on the items to be collected after item b_i and the corresponding configuration is not necessarily a sub-configuration of a feasible configuration for the whole sequence S_r. Consequently, this configuration is called, hereafter, a locally-optimal configuration for item b_i (even if $b_i \in \mathcal{H}$). When condition $(c1)$ is not met for b_i, i.e., $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) < k$, then there remain in any locally-optimal configuration for b_i exactly $k - l(i) \bmod k$ empty positions above a fragile item to which no non-fragile item can be assigned. Therefore, the maximum number of non-fragile items that can be loaded after item b_i is equal to $mk - l(i) - (k - l(i) \bmod k) = U(i)$, where $mk - l(i)$ is the total number of empty positions. On the other hand, when condition $(c1)$ holds for b_i, we can show that either 1) item b_i has no impact on the number of additional non-fragile items that can be loaded, or 2) the corresponding upper bound is equal to $mk - l(i)$ and, thus, redundant with the vehicle capacity constraint. In both cases, there is no need to compute the upper bound.

To illustrate these different cases, let us consider the following example. Let $m = 4$, $k = 3$, and $S_r = (b_1^{\mathcal{L}}, b_2^{\mathcal{H}}, b_3^{\mathcal{L}}, b_4^{\mathcal{L}}, b_5^{\mathcal{L}}, b_6^{\mathcal{L}}, b_7^{\mathcal{H}}, b_8^{\mathcal{L}}, b_9^{\mathcal{L}})$, where the upper index indicated if the item belongs to $\mathcal{L}$ or $\mathcal{H}$. Table 1 displays relevant values for each item $b_i^{\mathcal{L}}$: the left-hand side of condition $(c1)$, i.e., $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i))$; the upper bound $U(i)$ of condition $(c2)$ whenever condition $(c1)$ does not hold; the upper bound $mk - l(i)$ induced by the vehicle capacity; and the exact maximum number of non-fragile items that can be loaded after item $b_i^{\mathcal{L}}$ ($max^{\mathcal{H}}(i)$). The latter is computed as:

$$max^{\mathcal{H}}(i) = \min\{mk - l(i), \min_{j \in B_i}\{U(j) - (a^{\mathcal{H}}(i) - a^{\mathcal{H}}(j))\}\}, \tag{6}$$

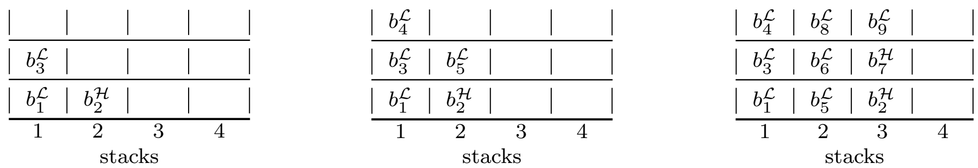
where B_i is the subset of fragile item indices $j \in \{1, \ldots, i\}$ such that $a^{\mathcal{H}}(j) + F(a^{\mathcal{L}}(j)) \geq k$. This maximum is obtained by considering the upper bounds induced by the vehicle capacity and by the fragile items $b_j^{\mathcal{L}}$, $j < i$, for which condition $(c1)$ does not hold. In the latter case, the corresponding upper bound $U(j)$ is adjusted by subtracting the number of non-fragile items loaded between $b_j^{\mathcal{L}}$ and $b_i^{\mathcal{L}}$.

Let us discuss the information reported in Table 1 for some specific items $b_i^{\mathcal{L}}$. For $i = 1$, the values are obvious because item $b_1^{\mathcal{L}}$ must be assigned to a stack, leaving the other three stacks (i.e., 9 positions)

Table 1: Values for items in $\mathcal{L}$ for an example with $m = 4$ and $k = 3$

i	$a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i))$	$U(i)$	$mk - l(i)$	$max^{\mathcal{H}}(i)$
1	1	9	11	9
3	3	-	9	8
4	4	-	8	8
5	2	6	7	6
6	3	-	6	6
8	5	-	4	4
9	3	-	3	3

to potentially load non-fragile items. For $i = 3$, condition $(c1)$ is satisfied $(a^{\mathcal{H}}(3) + F(a^{\mathcal{L}}(3)) = 3 \geq 3)$. In this case, $b_3^{\mathcal{L}}$ can be positioned on top of a fragile item $(b_1^{\mathcal{L}})$ in the locally-optimal configuration for $b_2^{\mathcal{H}}$ (see Figure 2a) and, therefore, does not impact the number of non-fragile items that can still be loaded. Note that $max^{\mathcal{H}}(3) = 8$ stems from the upper bound $U(1) = 9$ from which item $b_2^{\mathcal{H}}$ has been subtracted. For $i = 5$, condition $(c1)$ is not satisfied $(a^{\mathcal{H}}(5) + F(a^{\mathcal{L}}(5)) = 2 < 3)$ and the locally-optimal configuration (see Figure 2b) yields $U(5) = 6$ as at least one $(= k - l(5) \bmod k)$ of the seven $(= mk - l(5))$ unoccupied positions cannot be assigned to a non-fragile item (e.g., the empty position in stack 2 in Figure 2b). Note that item $b_5^{\mathcal{L}}$ induces a drop of two (from $max^{\mathcal{H}}(4) = 8$ to $max^{\mathcal{H}}(5) = 6$) in the exact maximum number of non-fragile items that can still be loaded. Finally, for $i = 9$, condition $(c1)$ holds. In this case, the locally-optimal configuration for $b_9^{\mathcal{L}}$ (see Figure 2c) provides an upper bound that is equal to $mk - l(9) = 3$ and is, thus, redundant with the vehicle capacity bound.



(a) For item $b_3^{\mathcal{L}}$ **(b) For item $b_5^{\mathcal{L}}$** **(c) For item $b_9^{\mathcal{L}}$**

Figure 2: Locally-optimal configurations for some items of the example

To prove Proposition 1, we proceed by induction on the number of stacks m. To make explicit the dependence of the proposition on the number of stacks, we will use the notation $U(i, m)$ instead of $U(i)$. The main line of the proof consists first in showing that the basis of the induction $P(m)$ with $m = 1$ is true. In particular we show that the feasibility and characterization conditions are equivalent: $Feas(1) \iff Char(1)$. We then prove the induction step $P(m) \implies P(m + 1)$. In particular, by assuming that the characterization is true for a generic number m of stacks, i.e., $Feas(m) \iff Char(m)$, we show that $Feas(m + 1) \impliedby Char(m + 1)$ as well as $Feas(m + 1) \implies Char(m + 1)$, thus proving that $Feas(m + 1) \iff Char(m + 1)$ and completing the argument.

P(1) is true

Proof. To prove that $P(1)$ is true, we need to show that $Feas(1) \iff Char(1)$. Observe that the case with no fragile item is trivial because all routes are feasible with respect to the fragility constraint and, in this case, the characterization requires no condition to be verified. We can then concentrate on routes including fragile items $(a^{\mathcal{L}}(s) > 0)$. We prove the sufficiency part first and the necessity second.

Feas(1) $\implies$ Char(1) Feasibility implies that non-fragile items are not stacked on top of fragile ones, i.e., $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) = 0$ for all $b_i \in \mathcal{L}$. For all $b_i \in \mathcal{L}$ such that $l(i) < k$, we have that $U(i, 1) = 0$, and condition $(c2)$ is satisfied. Furthermore, if there exists $b_i \in \mathcal{L}$ such that $l(i) = k$, then $F(a^{\mathcal{L}}(i)) = a^{\mathcal{L}}(i)$ and $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) = a^{\mathcal{H}}(i) + a^{\mathcal{L}}(i) = l(i) \geq k$ and condition (c1) is verified.

Feas(1) $\impliedby$ Char(1) We prove the following counterpositive version of this statement:

If sequence S_r is infeasible with respect to the fragility constraint, then there exists an item $b_i \in \mathcal{L}$ such that

(c3) $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) < k$

and

(c4) $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) > U(i,1)$.

If S_r is infeasible, then a non-fragile item is collected after a fragile one and there exists $i \in \{1,\ldots,s\}$ such that $b_i \in \mathcal{L}$ and $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) > 0$. Furthermore, $l(i) < k$ because an item is loaded after b_i. Consequently, $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) = a^{\mathcal{H}}(i) + a^{\mathcal{L}}(i) = l(i) < k$, thus condition $(c3)$ is satisfied. Also, $U(i,1) = 0$, and condition $(c4)$ follows.

□

P(m) $\implies$ P(m+1)

Proof. Here we prove that $P(m+1)$ is true when assuming that $P(m)$ is true, and in particular we need to prove that $Feas(m+1) \iff Char(m+1)$. We do this by proving the necessity condition first and sufficiency second.

$\mathbf{Feas(m+1)} \impliedby \mathbf{Char(m+1)}$

We start by considering a route r collecting at most $(m+1)k$ items and complying with $Char(m+1)$. We build a suitable subsequence $\bar{S}_r$ of at most mk items and we show that $\bar{S}_r$ complies with $Char(m)$. By the induction hypothesis, this implies that $\bar{S}_r$ is a feasible sequence. The feasibility of S_r will be trivially obtained by construction.

We distinguish two cases:

1. $a^{\mathcal{L}}(s) \geq k$. In this case, route r collects enough fragile items to be able to complete a stack made of fragile items only. Consider the sequence $\bar{S}_r$ of at most mk items obtained from S_r by eliminating the first k fragile items. Formally, $\bar{S}_r = (b_i \mid b_i \in S_r$ and either $b_i \in \mathcal{H}$ or $b_i \in \mathcal{L}$ and $a^{\mathcal{L}}(i) > k)$. We use bars to indicate that a given entity refers to $\bar{S}_r$; for example, $\bar{a}_{\mathcal{H}}(i)$ is the number of non-fragile items collected in $\bar{S}_r$ until and including item b_i.
Given that the number of non-fragile items collected in S_r and $\bar{S}_r$ is the same and that exactly k fragile items have been deleted from S_r, for all $i \in \{1,\ldots,s\}$ with $b_i \in \bar{S}_r$, we have

$$\bar{a}^{\mathcal{H}}(i) = a^{\mathcal{H}}(i) \tag{7}$$
$$\bar{a}^{\mathcal{L}}(i) = a^{\mathcal{L}}(i) - k \tag{8}$$
$$\bar{l}(i) = l(i) - k. \tag{9}$$

To prove that $\bar{S}_r$ is feasible, we show that it satisfies $Char(m)$. To this scope, let us consider a generic fragile item b_i in $\bar{S}_r$. Because we assume that $Char(m+1)$ holds for S_r, b_i satisfies condition $(c1)$ or condition $(c2)$ for S_r, yielding the following two cases.

i) Condition $(c1)$ holds, i.e., $a^{\mathcal{H}}(i) + F((a^{\mathcal{L}}(i))) \geq k$. Equations (7) and (8) imply $\bar{a}^{\mathcal{H}}(i) + F((\bar{a}^{\mathcal{L}}(i))) \geq k$, hence b_i satisfies condition $(c1)$ for $\bar{S}_r$.

ii) Condition $(c1)$ holds, i.e., $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) \leq U(i, m+1)$. To prove the statement, we show that this implies $\bar{a}^{\mathcal{H}}(s) - \bar{a}^{\mathcal{H}}(i) \leq \bar{U}(i,m)$, hence b_i satisfies condition $(c2)$ for $\bar{S}_r$.

$$\begin{aligned}
\bar{a}^{\mathcal{H}}(s) - \bar{a}^{\mathcal{H}}(i) &= a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) && (10)\\
&\leq U(i, m+1) && (11)\\
&= (m+1)k - (l(i) - l(i) \bmod k + k) && (12)\\
&= mk - (\bar{l}(i) - \bar{l}(i) \bmod k + k) && (13)\\
&= \bar{U}(i,m), && (14)
\end{aligned}$$

where Equations (10) and (13) hold given (7) and (9), respectively.

As a consequence of the induction hypothesis, the sequence $\bar{S}_r$ is feasible when considering m stacks. Then, for $m+1$ stacks, S_r is also feasible because it is always possible to stack the first k fragile items (that were put aside when building $\bar{S}_r$) in a dedicated stack.

2. $a^{\mathcal{L}}(s) < k$. We use a similar technique as for the previous case. Because it is not possible to fill a stack with fragile items only, the strategy consists in building a dedicated stack with a number of non-fragile items collected before the first fragile item, and with all fragile items. In case the number of non-fragile items collected before the first fragile item plus the total number of fragile items is less than k, the dedicated stack will be incomplete.

   Let f be the index in S_r of the first fragile item. Consider the subsequence $\bar{S}_r$ obtained from S_r by eliminating the first $min\{a^{\mathcal{H}}(f), k - a^{\mathcal{L}}(s)\}$ non-fragile items and $a^{\mathcal{L}}(s)$ fragile item. Because $\bar{S}_r$ is only made of non-fragile items, it is certainly feasible with respect to the fragility constraint. However, we need to ensure the capacity constraint of $\bar{S}$, i.e., that $\bar{l}(s) \leq mk$ is fulfilled. We distinguish two cases:

   - $k - a^{\mathcal{L}}(s) \leq a^{\mathcal{H}}(f)$. In this case, the construction eliminates exactly k items and $\bar{l}(s) = l(s) - k \leq mk$.
   - $k - a^{\mathcal{L}}(s) > a^{\mathcal{H}}(f)$. In this case, the construction eliminates $a^{\mathcal{H}}(f) + a^{\mathcal{L}}(s)$ items. Hence,

$$\begin{aligned}
\bar{l}(s) &= l(s) - a^{\mathcal{H}}(f) - a^{\mathcal{L}}(s) && (15)\\
&= a^{\mathcal{H}}(s) - a^{\mathcal{H}}(f) && (16)\\
&\leq U(f, m+1) && (17)\\
&= (m+1)k - (l(f) - l(f) \bmod k + k) && (18)\\
&= mk. && (19)
\end{aligned}$$

   Equation (16) holds because $l(s) = a^{\mathcal{H}}(s) + a^{\mathcal{L}}(s)$. Inequality (17) is due to the hypothesis that S_r complies with $Char(m+1)$ and, in particular, with condition $(c2)$ for b_f because $k - a^{\mathcal{L}}(s) > a^{\mathcal{H}}(s)$ implies $a^{\mathcal{H}}(f) + a^{\mathcal{L}}(f) < k$ and, thus, that condition $(c1)$ $(a^{\mathcal{H}}(f) + F(a^{\mathcal{L}}(f)) \geq k)$ does not hold. Finally, Equation (19) holds because $l(f) = a^{\mathcal{H}}(f) + a^{\mathcal{L}}(f) < k$.

   Consequently, $\bar{S}_r$ is feasible and S_r is also feasible, because it is always possible to build a dedicated stack with the eliminated items. This completes the proof of the necessity part of the statement.

$\mathbf{Feas(m+1) \implies Char(m+1)}$

We distinguish two cases:

1. $a^{\mathcal{L}}(s) \geq k$. Here, we use some of the constructions developed in the above necessity proof. We start with a sequence S_r of at most $(m+1)k$ items, which we assume to be feasible. Then, we construct a suitable sequence $\bar{S}_r$ of at most mk items and show that it is feasible. By using the induction hypothesis ($Feas(m) \iff Char(m)$), we then prove that S_r complies with $Char(m+1)$.

   As in the necessity proof, let $\bar{S}_r$ be the subsequence obtained by eliminating the first k fragile items from S_r. Because we assume that S_r is feasible and it is always possible to put the first k fragile items in a dedicated stack without affecting the feasibility of S_r, the subsequence $\bar{S}_r$ is also feasible and, by the induction hypothesis, $\bar{S}_r$ complies with $Char(m)$. We also observe that relations (7)–(9) hold by construction.

   To prove that S_r complies with $Char(m+1)$, we first consider the fragile items b_i belonging to both S_r and $\bar{S}_r$ and observe that, by the same reasoning done for the necessity proof, they all satisfy at least one of the conditions of $Char(m+1)$. In fact, for a given $i \in \{1, \ldots, s\}$ such that $b_i \in \mathcal{L} \cap \bar{S}_r$, we have two possible cases:

   i) $\bar{a}^{\mathcal{H}}(i) + F(\bar{a}^{\mathcal{L}}(i)) \geq k$. Then Equations (7) and (8) imply $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) \geq k$, hence b_i satisfies condition $(c1)$ for S_r.

ii) $\bar{a}^{\mathcal{H}}(s) - \bar{a}^{\mathcal{H}}(i) \leq \bar{U}(i,m)$. In this case, we get

$$\begin{aligned} a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) &= \bar{a}^{\mathcal{H}}(s) - \bar{a}^{\mathcal{H}}(i) \\ &\leq \bar{U}(i,m) \\ &= mk - (\bar{l}(i) - \bar{l}(i) \bmod k + k) \\ &= (m+1)k - (l(i) - l(i) \bmod k + k) \\ &= U(i, m+1). \end{aligned}$$

Hence, condition $(c2)$ holds for S_r.

Now, let us verify that one of the conditions of $Char(m+1)$ also holds for the first k fragile items belonging to S_r only. If b_i is such that $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) \geq k$, condition $(c1)$ of $Char(m+1)$ is fulfilled. Otherwise, we must have $a^{\mathcal{H}}(i) + a^{\mathcal{L}}(i) < k$ (because $a^{\mathcal{L}}(i) \leq k$) and, consequently, $l(i) < k$. Thus,

$$a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) \leq mk = U(i, m+1), \tag{20}$$

where the inequality arises from the fact that $a^{\mathcal{L}}(s) \geq k$ and the equality holds because $l(i) < k$.

2. $a^{\mathcal{L}}(s) < k$. In this case we can prove the statement directly, without employing the induction hypothesis. We have that $F(a^{\mathcal{L}}(i)) = a^{\mathcal{L}}(i)$ for all $i \in \{1, \ldots, s\}$ with $b_i \in \mathcal{L}$. Consequently, if item $b_i \in \mathcal{L}$ satisfies $a^{\mathcal{H}}(i) + a^{\mathcal{L}}(i) \geq k$, condition $(c1)$ of $Char(m+1)$ is fulfilled. Otherwise, $a^{\mathcal{H}}(i) + a^{\mathcal{L}}(i) < k$, implying that $l(i) < k$ and $U(i, m+1) = mk$. In this case, condition $(c2)$ writes as $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) \leq mk$. To prove that this condition is met, we start by observing that feasibility implies $a^{\mathcal{H}}(s) \leq mk + \min\{a^{\mathcal{H}}(f), k - a^{\mathcal{L}}(s)\}$, where f is again the index of the first fragile item in S_r.

   We consider two cases:

   i) $a^{\mathcal{H}}(f) < k - a^{\mathcal{L}}(s)$. In this case, $a^{\mathcal{H}}(s) \leq mk + a^{\mathcal{H}}(f)$ and we deduce

$$\begin{aligned} a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) &\leq a^{\mathcal{H}}(s) - a^{\mathcal{H}}(f) \\ &\leq mk + a^{\mathcal{H}}(f) - a^{\mathcal{H}}(f) = mk. \end{aligned}$$

   Therefore, condition $(c2)$ holds.

   ii) $a^{\mathcal{H}}(f) \geq k - a^{\mathcal{L}}(s)$. In this case, $a^{\mathcal{H}}(s) \leq mk + k - a^{\mathcal{L}}(s)$ and we find

$$\begin{aligned} a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) &\leq a^{\mathcal{H}}(s) - a^{\mathcal{H}}(f) \\ &\leq mk + k - a^{\mathcal{L}}(s) - a^{\mathcal{H}}(f) \\ &\leq mk, \end{aligned}$$

   where the last inequality comes from the assumption that $a^{\mathcal{H}}(f) \geq k - a^{\mathcal{L}}(s)$, i.e., $k - a^{\mathcal{L}}(s) - a^{\mathcal{H}}(f) \leq 0$. Consequently, condition $(c2)$ is also satisfied.

This completes the proof of the sufficiency part of the statement. □

The characterization provided in Proposition 1 requires that conditions $(c1)$ and $(c2)$ are satisfied for all fragile items b_i in S_r. The following proposition ensures that, if the characterization conditions are satisfied for a suitable subset of fragile items, they are satisfied for all fragile items. In fact, it turns out that it is sufficient to check the conditions of the characterization only for $b_i \in \mathcal{L}$ such that $i = s$ or $b_{i+1} \notin \mathcal{L}$. In particular, when route r visits a customer $i \in \mathcal{L}$ with demand q_i, there is no need to consider these items individually. Moreover, if r visits several customers in $\mathcal{L}$ consecutively, it is sufficient that the last item of the last of these customers verifies the conditions.

Proposition 2 *Let $i \in \{1, \ldots, s-1\}$ and $b_i, b_{i+1} \in \mathcal{L}$. If*

$$a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) < k \quad \textit{and} \quad a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) > U(i)$$

(i.e., conditions $(c1)$ *and* $(c2)$ *are not satisfied for* b_i*), then*

$$a^{\mathcal{H}}(i+1) + F(a^{\mathcal{L}}(i+1)) < k \quad \textit{and} \quad a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i+1) > U(i+1)$$

(i.e., they are neither satisfied for b_{i+1}*).*

Proof. Let us denote the last two conditions $(c5)$ and $(c6)$. Observe that $a^{\mathcal{H}}(i+1) = a^{\mathcal{H}}(i)$, $a^{\mathcal{L}}(i+1) = a^{\mathcal{L}}(i) + 1$ and $l(i+1) = l(i) + 1$. Furthermore, it is easy to prove that $U(i+1) \leq U(i)$.

We get that $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i+1) = a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) > U(i) \geq U(i+1)$, which shows that condition $(c6)$ is always satisfied for b_{i+1}.

Let us now consider two cases:

1. If $a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) < k-1$, then

$$\begin{aligned} a^{\mathcal{H}}(i+1) + F(a^{\mathcal{L}}(i+1)) &= a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)+1) \\ &= a^{\mathcal{H}}(i) + F(a^{\mathcal{L}}(i)) + 1 \\ &< k, \end{aligned}$$

   where the second equality is valid because $F(a^{\mathcal{L}}(i)) < k-1$. Thus, condition $(c5)$ is also satisfied.
2. If $a^{\mathcal{H}}(i)+F(a^{\mathcal{L}}(i)) = k-1$, then $l(i) = a^{\mathcal{H}}(i)+a^{\mathcal{L}}(i) = k-1-F(a^{\mathcal{L}}(i))+a^{\mathcal{L}}(i)$. Because $a^{\mathcal{L}}(i) > 0$, we have that $a^{\mathcal{L}}(i) - F(a^{\mathcal{L}}(i)) = \psi k$, where ψ is a nonnegative integer. Thus, $l(i) = k-1+\psi k$, $l(i) \bmod k = k-1$ and $U(i) = mk - l(i) - 1$. The total number of items in the sequence is at least $l(i+1) + a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i+1) = l(i) + 1 + a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i) > l(i) + 1 + U(i) = mk$, which is not possible given the vehicle capacity. Consequently, case 2 is not possible and the proof is complete.

□

The labeling algorithm described in Section 4.1.2 largely stands on Propositions 1 and 2.

To conclude this section, we prove two other propositions concerning specific cases. Even if the stated results are not exploited in the proposed BPC algorithm, they are useful to understand when it is more difficult to satisfy the fragility constraint. Furthermore, they served to design our test instances and to analyze the obtained computational results.

Proposition 3 *Consider a route* $r = (v_0, v_1, \ldots, v_p, v_{p+1})$*. If* $\sum_{j=1}^{p} q_{v_j} \leq Q - k$*, then* r *satisfies the fragility constraint.*

Proof. Assume that $\sum_{j=1}^{p} q_{v_j} \leq Q - k$. From Propositions 1 and 2, we can prove that r is fragility-feasible by showing that, for the last item b_{i_j} of every customer $v_j \in \mathcal{L}$, condition $(c2)$ is satisfied. Let b_s be the last item of a customer v_p. If $j = p$, then $i_j = s$, $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i_j) = 0$, and $U(i_j, m) \geq 0$, implying condition $(c2)$. For $j < p$, we get

$$\begin{aligned} a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i_j) &\leq \sum_{j'=j+1}^{p} q_{v_{j'}} \\ &\leq Q - k - \sum_{j'=1}^{j} q_{v_{j'}} \\ &= mk - k - l(i_j) \\ &\leq mk - k - l(i_j) + l(i_j) \bmod k \\ &= U(i_j, m), \end{aligned}$$

where the first inequality comes from the fact that some customers $v_{j'}$, $j' \in \{j+1, \ldots, p\}$, might belong to set $\mathcal{L}$ and the second inequality from the assumption. Therefore, condition $(c2)$ holds. □

Proposition 3 indicates that routes must be sufficiently filled to be fragility-infeasible. For example, if we consider $m = 12$ stacks of height $k = 4$, then only routes with a total load of 45 or more (i.e., with a filling rate exceeding 91.7%) can violate the fragility constraint. Next, we discuss the case $k = 2$.

Proposition 4 *Let $r = (v_0, v_1, \dots, v_p, v_{p+1})$ be a route that is feasible with respect to the capacity constraint and let $k = 2$. Route r does not satisfy the fragility constraint if and only if $p \geq 2$, $\sum_{j=1}^{p} q_{v_j} = Q$ and there exists an index $j^* \in \{1, \dots, p-1\}$ such that $v_j \in \mathcal{L}$ for all $j \in \{1, \dots, j^*\}$, $v_j \in \mathcal{H}$ for all $j \in \{j^*+1, \dots, p\}$ and $\sum_{j=1}^{j^*} q_{v_j}$ is odd.*

Proof. ($\Longrightarrow$) Assume that r does not satisfy the fragility constraint. From Propositions 1 and 2, it means that there exists an index $j^* \in \{1, \dots, p\}$ such that $v_{j^*} \in \mathcal{L}$ and both conditions $(c1)$ and $(c2)$ are not met for the last item $b_{i_{j^*}}$ of v_{j^*}. Because $k = 2$ and $F(a^{\mathcal{L}}(i_{j^*})) \in \{1, 2\}$, condition $(c1)$ is not satisfied if and only if $a^{\mathcal{H}}(i_{j^*}) = 0$ and $F(a^{\mathcal{L}}(i_{j^*})) = 1$. This means that $v_j \in \mathcal{L}$ for all $j \in \{1, \dots, j^*\}$ and $\sum_{j=1}^{j^*} q_{v_j}$ is odd. Moreover, if b_s denotes the last item of v_p, condition $(c2)$ writes as $a^{\mathcal{H}}(s) - 0 \leq Q - (a^{\mathcal{L}}(i_{j^*}) - 1 + 2)$ because $l(i_{j^*}) = a^{\mathcal{L}}(i_{j^*})$ which is odd. This condition is thus violated if $a^{\mathcal{H}}(s) \geq Q - a^{\mathcal{L}}(i_{j^*})$, i.e., if $a^{\mathcal{H}}(s) + a^{\mathcal{L}}(i_{j^*}) = Q$ or, equivalently, $\sum_{j=1}^{p} q_{v_j} = Q$ and $v_j \in \mathcal{H}$ for all $j \in \{j^*+1, \dots, p\}$.

($\Longleftarrow$) Assume that $p \geq 2$, $\sum_{j=1}^{p} q_{v_j} = Q$ and there exists an index $j^* \in \{1, \dots, p-1\}$ such that $v_j \in \mathcal{L}$ for all $j \in \{1, \dots, j^*\}$, $v_j \in \mathcal{H}$ for all $j \in \{j^*+1, \dots, p\}$ and $\sum_{j=1}^{j^*} q_{v_j}$ is odd. Denote by $b_{i_{j^*}}$ the last item at customer v_{j^*}. Let us show that both conditions $(c1)$ and $(c2)$ of Proposition 1 do not hold for $b_{i_{j^*}}$, indicating that route r is fragility-infeasible. From the assumption, we can easily deduce that $a^{\mathcal{H}}(i_{j^*}) = 0$ and $F(a^{\mathcal{L}}(i_{j^*})) = 1$. Thus, $(c1)$ is violated. Furthermore, we find that $a^{\mathcal{H}}(s) + a^{\mathcal{L}}(i_{j^*}) = \sum_{j=1}^{p} q_{v_j} = Q$ and $l(i_{j^*}) = a^{\mathcal{L}}(i_{j^*})$. Consequently, $a^{\mathcal{H}}(s) - a^{\mathcal{H}}(i_{j^*}) = Q - a^{\mathcal{L}}(i_{j^*}) > U(i_{j^*}) = Q - (a^{\mathcal{L}}(i_{j^*}) - 1 + 2)$, proving that condition $(c2)$ does not hold either. □

Proposition 4 describes explicitly the routes that do not satisfy the fragility constraint when $k = 2$. In particular, it shows that, if a customer in $\mathcal{H}$ is visited before a customer in $\mathcal{L}$, then the route is necessary fragility-feasible. Indeed, it gives the opportunity to start a stack with a non-fragile item and fill it with a fragile item if needed. This principle, which is expressed by condition $(c1)$, is also valid for larger stack heights k. From it, we can deduce that routes visiting customers in $\mathcal{H}$ with sufficiently large demands before customers in $\mathcal{L}$ have high chances to be fragility-feasible. This may not be true for a large stack height k but, in practice, k is not too large.

4 The branch-price-and-cut framework

BPC is a variant of the branch-and-bound algorithm where the lower bounds at the nodes of the search tree are computed via column generation. Lower bounds are then strengthened by the dynamic generation of valid inequalities. In Section 4.1, we present the proposed column generation algorithm for solving the linear relaxation of the F-VRPTW formulation (2)–(4). In Section 4.2, we describe some acceleration strategies. Cutting planes and branching strategies are then discussed in Sections 4.3 and 4.4, respectively. Finally, an alternative BPC algorithm which does not exploit the fragility-feasible route characterization is introduced in Section 4.5. This algorithm will be used as a benchmark against the proposed BPC algorithm.

4.1 Column generation

At each node of the search tree, the BPC algorithm must compute a lower bound by solving the linear relaxation of (2)–(4) possibly augmented by the constraints implied by the branching decisions or valid inequalities previously added. However, due to the extremely large number of variables in (2)–(4), this linear problem cannot be solved directly and, consequently, an iterative column generation procedure needs to be invoked (see, e.g., Lübbecke and Desrosiers 2005). A column generation iteration first solves a Restricted Master Problem (RMP) where only a small subset $\Omega' \subset \Omega$ of feasible routes is considered. Let $\bar{x}$ be the solution of the RMP. The iteration then proceeds by solving a pricing problem to verify the optimality of $\bar{x}$ for the whole linear relaxation. When the optimality check fails, a new set of routes (identified by the pricing problem) is added to the RMP, and the process iterates. Otherwise, the column generation procedure stops as the current RMP solution yields a valid lower bound.

In Section 4.1.1, we define the pricing problem, which turns out to be a variant of the ESPPRC. This problem is solved using the labeling algorithm described in Section 4.1.2.

4.1.1 Pricing problem formulation

To verify the optimality of the current RMP solution in the column generation procedure, the pricing problem searches for routes with a negative reduced cost. Let π_i for all $i \in \mathcal{N}_c$ be the dual variables associated with constraints (3). The reduced cost of a route can be expressed as follows:

$$\bar{c}_r = c_r - \sum_{i \in \mathcal{N}_c} a_{ir} \pi_i. \tag{21}$$

Thus, the pricing problem minimizes (21) over the set of all feasible routes Ω.

Any route $r \in \Omega$ can be represented as a path in network G. By combining expressions (1) and (21), it is possible to express the reduced cost $\bar{c}_r$ of a route $r = (v_0, v_1, \ldots, v_p, v_{p+1})$ as the sum of the contributions of its arcs: $\bar{c}_r = \sum_{i=1}^{p} \bar{c}_{v_i, v_{i+1}}$, where

$$\bar{c}_{v_i, v_{i+1}} = \begin{cases} c_{v_i, v_{i+1}} & \text{if } v_i = 0, \\ c_{v_i, v_{i+1}} - \pi_{v_i} & \text{otherwise.} \end{cases} \tag{22}$$

This property can be suitably used to express the column generation pricing problem as an arc-flow formulation which is presented in Appendix A. This formulation is, however, a non-linear integer program that is hard to solve via state-of-the-art solvers. On the other hand, we observe that the pricing problem corresponds to an ESPPRC, where the resources ensure route feasibility with respect to the time windows, the vehicle capacity, and the fragility constraint.

4.1.2 Labeling algorithm

In the vehicle routing literature, the ESPPRCs are frequently solved by dynamic programming, which is usually implemented by means of a labeling algorithm (Irnich and Desaulniers 2005). In our specific problem setting, given that the fragility constraint is dealt with in the pricing problem, the labeling algorithm is potentially very challenging. In fact, given a route visiting a set of customers in a specific order, there are many possible ways to position the freight in the vehicle. Although some of these configurations may be infeasible, there could still exist a large number of feasible loading configurations. A potential labeling algorithm would create one new label for each feasible configuration or, if clever modeling is used to avoid symmetry between the identical stacks as in Cherkesly et al. (2016), one new label for each feasible stack-anonymous configuration. However, both intuition and preliminary computational tests for the simplest case with stacks of height $k = 2$ suggest that this method is not efficient due to the large number of generated labels (Altman 2017).

We then directed our research efforts towards the theoretical results of Propositions 1 and 2. In particular, the characterization of a fragility-feasible route enables us to carry information about the *existence* of a feasible loading configuration for a given partial route. In practice, as long as at least one feasible configuration exists, our labeling algorithm extends partial routes. However, no attempt is made to build a specific configuration, which can be trivially retrieved a posteriori.

A labeling algorithm (see Irnich and Desaulniers 2005) uses labels to represent partial paths (routes) in a network. Paths are enumerated by extending recursively an initial label E_0 from the source node towards the destination. Labels are extended according to extension functions. A dominance rule is applied to eliminate partial routes that cannot yield complete optimal routes. These algorithmic components are described next.

Label definition. A partial route $r = (0, \dots, i)$, $i \in \mathcal{N} \cup \{0\}$, is encoded by a label $E_r = (C_r, [V_r^l]_{l \in N_c}, T_r, L_r, A_r^{\mathcal{H}}, A_r^{\mathcal{L}}, X_r)$ with the following $n + 6$ components:

- one component C_r accounting for the reduced cost;
- n binary components V_r^l, $l \in \mathcal{N}_c$, indicating whether or not customer node l is *unreachable*, i.e., it cannot be visited in any feasible extension of route r because it has already been visited or because its time window or the vehicle capacity cannot be satisfied;
- one component T_r indicating the (earliest) service starting time at node r;
- one component L_r accounting for total collected load;
- two components $A_r^{\mathcal{H}}$ and $A_r^{\mathcal{L}}$ providing the total number of non-fragile and fragile items collected, respectively;
- one component X_r indicating the maximum number of non-fragile items that can still be collected according to Propositions 1 and 2.

The initial label at node 0 is $E_{r_0} = (0, [0]_{l \in \mathcal{N}_c}, 0, 0, 0, 0, Q)$, where partial route $r_0 = (0)$.

Label extension functions. Given a partial route $r = (0, \dots, i)$, $i \in \mathcal{N} \setminus \{n + 1\}$, with label $E_r = (C_r, [V_r^l]_{l \in N_c}, T_r, L_r, A_r^{\mathcal{H}}, A_r^{\mathcal{L}}, X_r)$, it can be extended along an arc $(i, j) \in \mathcal{A}$ using the following label extension functions to yield a new partial route $r' = (0, \dots, i, j)$ represented by the label $E_{r'} = (C_{r'}, [V_{r'}^l]_{l \in \mathcal{N}_c}, T_{r'}, L_{r'}, A_{r'}^{\mathcal{H}}, A_{r'}^{\mathcal{L}}, X_r)$:

$$C_{r'} = C_r + \bar{c}_{ij} \tag{23}$$

$$T_{r'} = \begin{cases} \max\{T_r + t_{ij}, \underline{w}_j\} & \text{if } j \neq n+1 \\ T_r + t_{ij} & \text{otherwise} \end{cases} \tag{24}$$

$$L_{r'} = L_r + q_j \tag{25}$$

$$A_{r'}^{\mathcal{H}} = \begin{cases} A_r^{\mathcal{H}} + q_j & \text{if } j \in \mathcal{H} \\ A_r^{\mathcal{H}} & \text{otherwise} \end{cases} \tag{26}$$

$$A_{r'}^{\mathcal{L}} = \begin{cases} A_r^{\mathcal{L}} + q_j & \text{if } j \in \mathcal{L} \\ A_r^{\mathcal{L}} & \text{otherwise} \end{cases} \tag{27}$$

$$V_{r'}^l = \begin{cases} V_r^l + 1 & \text{if } j = l \\ V_r^l & \text{if } j = n+1 \\ \max\{V_r^l, \mathcal{Z}_{jl}(T_{r'}, L_{r'})\} & \text{otherwise,} \end{cases} \qquad \forall l \in \mathcal{N}_c \tag{28}$$

$$X_{r'} = \begin{cases} \min\{X_r, Q - L_{r'}\} & \text{if } j \in \mathcal{L} \text{ and } A_{r'}^{\mathcal{H}} + F(A_{r'}^{\mathcal{L}}) \geq k \\ \min\{X_r, U_{r'}\} & \text{if } j \in \mathcal{L} \text{ and } A_{r'}^{\mathcal{H}} + F(A_{r'}^{\mathcal{L}}) < k \\ X_r - q_j & \text{otherwise,} \end{cases} \tag{29}$$

where $q_{n+1} = 0$, $\mathcal{Z}_{jl}(T_{r'}, L_{r'}) = 1$ if $T_{r'} + t_{jl} > \overline{w}_l$ or $L_{r'} + q_l > Q$ and 0 otherwise, and $U_{r'} = Q - (L_{r'} - L_{r'} \bmod k + k)$ is defined as in Proposition 1 (recall that $Q = mk$).

Extension function (29) relies on Proposition 1 and the vehicle capacity constraint to compute the maximum number of non-fragile items that can still be loaded after visiting a customer. In the first case, given that $A_{r'}^{\mathcal{H}} + F(A_{r'}^{\mathcal{L}}) \geq k$, the loading of the q_j fragile items at node j does not affect the maximum number of non-fragile items that can still be loaded in virtue of the fragility constraint. However, if the capacity constraint becomes binding, the term $Q - L_{r'}$ in the minimum function ensures that $X_{r'}$ does not exceed the residual available space (i.e., $X_{r'} \leq Q - L_{r'}$). The second case is similar but, when $A_{r'}^{\mathcal{H}} + F(A_{r'}^{\mathcal{L}}) < k$, Proposition 1 imposes a maximum number of non-fragile items $U_{r'}$. In this case, the capacity limit is implicit because $Q - L_{r'} > U_{r'}$. Finally, in the third case, because $j \in \mathcal{H}$ or $j = n + 1$, the extension function simply decreases the number of non-fragile items by q_j.

The obtained label $E_{r'}$ is declared feasible if $j = n + 1$ or if $V_{r'}^l \leq 1$ for all $l \in \mathcal{N}_c$, $T_{r'} \leq \overline{w}_j$, $L_{r'} \leq Q$, and $X_{r'} \geq 0$. Otherwise, it is discarded. Note that all these conditions except the last one can be verified before performing the label extension to a node $j \in \mathcal{N}_c$ by checking the value of V_r^j as follows. If $V_r^j = 1$, then at least one of these conditions will not be met and there is no need to compute $E_{r'}$. Otherwise, the extension must be performed and the condition $X_{r'} \geq 0$ must be checked to determine the feasibility of label $E_{r'}$.

Dominance rule. Let r_d, $d = 1, 2$, be two partial routes, both ending at the same node $i \in \mathcal{N}$ and represented by the labels $E_{r_d} = (C_{r_d}, [V_{r_d}^l]_{l \in \mathcal{N}_c}, T_{r_d}, L_{r_d}, A_{r_d}^{\mathcal{H}}, A_{r_d}^{\mathcal{L}}, X_{r_d})$, $d = 1, 2$. We say that E_{r_1} dominates E_{r_2} if

($c7$) any feasible (single- or multiple-arc) extension e of r_2 is also feasible for r_1, and

($c8$) for any such extension e, the inequality $C_{r_1 \oplus e} \leq C_{r_2 \oplus e}$ holds, where symbol $\oplus$ denotes the concatenation operator.

The dominated label E_{r_2} can then be discarded. However, when multiple labels dominate each other, one of them must be kept.

The dominance conditions ($c7$) and ($c8$) cannot be checked easily in practice. In general, sufficient conditions defining a so-called dominance rule are used instead. In particular, when all label extension functions are monotone (either non-decreasing or non-increasing), it is easy to provide such a dominance rule (see, e.g., Irnich and Desaulniers 2005). In our case, monotonicity does not hold for extension function (29). Consequently, the dominance rule that we introduce in the following proposition is more complex and will be proven.

Proposition 5 *Route r_1 dominates route r_2 if the following relations hold:*

$$C_{r_1} \leq C_{r_2} \tag{30}$$
$$T_{r_1} \leq T_{r_2} \tag{31}$$
$$L_{r_1} \leq L_{r_2} \tag{32}$$
$$V_{r_1}^l \leq V_{r_2}^l, \qquad \forall l \in \mathcal{N}_c \tag{33}$$
$$\min\{X_{r_1}, Q - L_{r_1} - (k - A_{r_1}^{\mathcal{H}} - 1)\} \geq Q - L_{r_2} \tag{34}$$

In the following, we use $\mathcal{R}(\sigma_1, \sigma_2)$ as a shorthand notation for the set of conditions (30)–(34) when $r_1 = \sigma_1$ and $r_2 = \sigma_2$. Before proving Proposition 5, we state and prove the following preliminary lemma.

Lemma 1 *If relations $\mathcal{R}(r_1, r_2)$ are true, then relations $\mathcal{R}(r_1 \oplus e, r_2 \oplus e)$ also hold for any extension e of r_2.*

Proof. We begin by showing that the statement holds true for any single-arc extension e made of a generic arc (i, j). To highlight that routes r_d, $d = 1, 2$, end at node i, we denote their labels by $E_{r_d} = (C_{di}, [V_{di}^l]_{l \in \mathcal{N}_c}, T_{di}, L_{di}, A_{di}^{\mathcal{H}}, A_{di}^{\mathcal{L}}, X_{di})$. Similarly, the labels for routes $r_d \oplus e$ are written $E_{r_d \oplus e} = (C_{dj}, [V_{dj}^l]_{l \in \mathcal{N}_c}, T_{dj}, L_{dj}, A_{dj}^{\mathcal{H}}, A_{dj}^{\mathcal{L}}, X_{dj})$.

Given that the extension functions (23)–(25) and (28) are non-decreasing, it is trivial to show that the statement holds for relations (30)–(33). We then concentrate on relation (34) for which we want to prove that:

$$\min\{X_{1i}, Q - L_{1i} - (k - A_{1i}^{\mathcal{H}} - 1)\} \geq Q - L_{2i} \implies \min\{X_{1j}, Q - L_{1j} - (k - A_{1j}^{\mathcal{H}} - 1)\} \geq Q - L_{2j}. \tag{35}$$

Let us assume that $\min\{X_{1i}, Q - L_{1i} - (k - A_{1i}^{\mathcal{H}} - 1)\} \geq Q - L_{2i}$. We distinguish three cases related to the extension of r_1, one for each case in (29).

- If $j \in \mathcal{H}$ or $j = n+1$, then

$$\begin{aligned}\min\{X_{1j}, Q - L_{1j} - (k - A^{\mathcal{H}}_{1j} - 1)\} &= \min\{X_{1i} - q_j, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1)\} \\ &\geq \min\{X_{1i}, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1)\} - q_j \\ &\geq Q - L_{2i} - q_j \\ &\geq Q - L_{2j},\end{aligned}$$

where the equality ensues from $X_{1j} = X_{1i} + q_j$ and $L_{1j} - A^{\mathcal{H}}_{1j} = L_{1i} + q_j - (A^{\mathcal{H}}_{1i} + q_j) = L_{1i} - A^{\mathcal{H}}_{1i}$, and the second inequality from the assumption.

- If $j \in \mathcal{L}$ and $A^{\mathcal{H}}_{1j} + F(A^{\mathcal{L}}_{1j}) \geq k$, then

$$\begin{aligned}\min\{X_{1j}, Q - L_{1j} - (k - A^{\mathcal{H}}_{1j} - 1)\} &= \min\{X_{1i}, Q - L_{1i} - q_j, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1) - q_j\} \\ &\geq \min\{X_{1i}, Q - L_{1i}, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1)\} - q_j \\ &= \min\{\min\{X_{1i}, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1)\}, Q - L_{1i}\} - q_j \\ &\geq \min\{Q - L_{2i}, Q - L_{1i}\} - q_j \\ &\geq Q - L_{2j},\end{aligned}$$

where the equality is derived from $X_{1j} = \min\{X_{1i}, Q - L_{1i} - q_j\}$ and $L_{1j} - A^{\mathcal{H}}_{1j} = L_{1i} - A^{\mathcal{H}}_{1i} + q_j$, the second inequality from the assumption, and the last from $L_{1i} \leq L_{2i}$.

- If $j \in \mathcal{L}$ and $A^{\mathcal{H}}_{1j} + F(A^{\mathcal{L}}_{1j}) < k$, then

$$\begin{aligned}\min\{X_{1j}, Q - L_{1j} - (k - A^{\mathcal{H}}_{1j} - 1)\} &= \min\{X_{1i}, U_j, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1) - q_j\} \\ &\geq \min\{X_{1i}, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1) - q_j\} \\ &\geq \min\{X_{1i}, Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1)\} - q_j \\ &\geq Q - L_{2i} - q_j \\ &\geq Q - L_{2j},\end{aligned}$$

where the equality arises from $X_{1j} = \min\{X_{1i}, U_j\}$ and $L_{1j} - A^{\mathcal{H}}_{1j} = L_{1i} - A^{\mathcal{H}}_{1i} + q_j$, the first inequality from $U_j \geq Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1) - q_j$ (as discussed next), and the third inequality from the assumption. Showing that $U_j = Q - L_{1i} - q_j - k + L_{1j} \bmod k \geq Q - L_{1i} - (k - A^{\mathcal{H}}_{1i} - 1) - q_j$ is equivalent to showing that $L_{1j} \bmod k \geq A^{\mathcal{H}}_{1i} + 1$. The latter is true because

$$L_{1j} \bmod k = (A^{\mathcal{H}}_{1j} + A^{\mathcal{L}}_{1j}) \bmod k = (A^{\mathcal{H}}_{1j} + F(A^{\mathcal{L}}_{1j})) \bmod k = A^{\mathcal{H}}_{1j} + F(A^{\mathcal{L}}_{1j}) \geq A^{\mathcal{H}}_{1i} + 1,$$

where the second equality stems from $A^{\mathcal{L}}_{1j} \geq 1$ (as $j \in \mathcal{L}$), yielding $A^{\mathcal{L}}_{1j} = F(A^{\mathcal{L}}_{1j})$, the third equality from $A^{\mathcal{H}}_{1j} + F(A^{\mathcal{L}}_{1j}) < k$, and the inequality from $A^{\mathcal{H}}_{1j} = A^{\mathcal{H}}_{1i}$ and $F(A^{\mathcal{L}}_{1j}) \geq 1$ (as $j \in L$).

This proves that the implication (35) holds and that the statement is true for single-arc extensions. By induction on the number of single-arc extensions, the lemma is also true for any extension e with an arbitrary number of arcs. □

Proof of Proposition 5. To prove this proposition, we show that, if relations $\mathcal{R}(r_1, r_2)$ are satisfied, then conditions ($c7$) and ($c8$) are also satisfied. Let e be a feasible extension of r_2 that ends at a node $j \in \mathcal{N}$. Denote by $E_{r_d \oplus e} = (C_{dj}, [V^l_{dj}]_{l \in \mathcal{N}_c}, T_{dj}, L_{dj}, A^{\mathcal{H}}_{dj}, A^{\mathcal{L}}_{dj}, X_{dj})$, $d = 1, 2$, the labels associated with $r_d \oplus e$. Because $r_2 \oplus e$ is feasible, we have $V^l_{2j} \leq 1$ for all $l \in \mathcal{N}_c$, $T_{2j} \leq \overline{w}_j$, $L_{2j} \leq Q$, and $X_{2j} \geq 0$. Assuming that relations $\mathcal{R}(r_1, r_2)$ hold, we deduce by Lemma 1 that

$$C_{1j} \leq C_{2j} \tag{36}$$

$$V^l_{1j} \leq V^l_{2j} \leq 1, \qquad \forall l \in \mathcal{N}_c \tag{37}$$

$$T_{1j} \leq T_{2j} \leq \overline{w}_j \tag{38}$$

$$L_{1j} \leq L_{2j} \leq Q \tag{39}$$

$$\min\{X_{1j}, Q - L_{1j} - (k - A^{\mathcal{H}}_{1j} - 1)\} \geq Q - L_{2j}. \tag{40}$$

From inequality (40), we can easily deduce that

$$X_{1j} \geq X_{2j} \geq 0 \tag{41}$$

because $Q - L_{2j} \geq X_{2j}$ by definition of X_{2j}. Therefore, inequalities (37)–(39) and (41) indicate that $r_1 \oplus e$ is feasible, i.e., that condition ($c7$) holds, whereas inequality (36) implies condition ($c8$). □

4.2 Acceleration strategies

Acceleration strategies play a key role for the development of efficient BPC algorithms. In this section we describe the adopted procedures, all of them geared towards improving the performance of the labeling algorithm in the column generation step.

4.2.1 Decremental state space relaxation

An efficient technique to generate negative reduced cost elementary paths is the decremental state space relaxation (DSSR) introduced independently by Boland et al. (2006) and Righini and Salani (2008). The pricing problem is initially solved without the label components $V^1, \ldots, V^n$, thus completely relaxing the elementarity requirements. If no negative reduced cost paths are found in this way, some of the corresponding label components are dynamically added, and the process is iterated until elementary paths with a negative reduced cost are found or the reduced cost of the shortest path is non-negative. As proposed in Desaulniers et al. (2008), at each iteration of the column generation, instead of initializing the algorithm with an empty set of visit-label components, we use the components generated in the previous iteration.

4.2.2 The ng-path relaxation

The technique described in this section consists in partially relaxing the elementarity requirements of partial routes. The adopted path relaxation, called ng-path, was introduced by Baldacci et al. (2011). With each node $i \in \mathcal{N}_c$, the approach associates a subset of nodes $N_i \subseteq \mathcal{N}_c$ such that $i \in N_i$ and $|N_i| \leq \Delta^0$, where Δ^0 is a predefined integer parameter. Typically, N_i contains the closest customers of i. Given a partial route $r = (v_0, \ldots, v_p)$, the subsets N_i allow us to define a new subset $\Pi(r)$ whose elements are prevented to be direct extension candidates for r. The subset $\Pi(r)$ is defined as $\Pi(r) = \{v_i \in r \mid v_i \in \bigcap_{l=i+1}^{p} N_{v_l},\ i = 1, \ldots, p-1\} \cup \{v_p\}$. The value of Δ^0 can influence the degree of elementarity of an ng-path. A small value of Δ^0 allows the ng-paths to contain many cycles, while $\Delta^0 = |\mathcal{N}_c|$ imposes elementarity. Allowing cycles generally makes the corresponding ESPPRC easier to solve, but the quality of the provided lower bound deteriorates. Hence the right choice of Δ^0 is key for the overall computational efficiency of the BPC algorithm.

Instead of choosing neighborhoods of the same size for all nodes, Contardo et al. (2015) propose to adjust them dynamically and individually in a DSSR fashion as follows. First, the size of each neighborhood is initially set to a relatively small value Δ^0 (10 in our tests). Column generation is then applied to solve a linear relaxation. If the computed linear relaxation solution contains routes with a cycle, neighbors are added to the nodes in these cycles in such a way that they become forbidden. The corresponding columns are then removed from the RMP and column generation is started over again. The process is repeated until the linear relaxation solution is free of cycles or when it is not possible to forbid the remaining cycles because a maximum of Δ^+ additional neighbors has already been added to a node ($\Delta^+ = 10$ for our tests).

To implement ng-path, the extension function of the label components $V^1, \ldots, V^n$ needs to be suitably modified (see Desaulniers et al. 2014; Costa et al. 2019, for a detailed description of the technique).

4.2.3 Heuristic dynamic programming

To rapidly obtain routes with negative reduced costs, we adopt graphs with reduced size. In particular, at each iteration, the labeling algorithm is first executed on a simplified graph G' containing only a subset $\mathcal{A}'$ of the arcs in $\mathcal{A}$. In case no route with negative reduced cost is found, the labeling algorithm is run again, but on the complete graph. As it is detailed in Desaulniers et al. (2008), the choice of the initial set of arcs $\mathcal{A}'$ depends on the current modified costs of the arcs (22). Thus, the set $\mathcal{A}'$ changes at every iteration, according to the dual values corresponding to RMP solution. More precisely, for every node $i \in \mathcal{N}$, all incoming arcs and all outgoing arcs are sorted according to their modified cost in two separated lists I_i and O_i. If an arc (i, j) is such that its position in both I_j and O_i is larger than a given threshold τ, then arc (i, j) is eliminated. In this study, we set $\tau = 10$.

4.3 Cutting planes

At a given node of the branch-and-bound tree, the column generation process provides a linear relaxation solution that may be fractional, as well as a valid lower bound on the value of the node. When this solution is fractional, it is possible to strengthen the obtained lower bound by looking for potential violated valid inequalities.

We consider two families of inequalities. The first was introduced in Jepsen et al. (2008) and is known as *subset-row* inequalities, which are Chvátal–Gomory rank-one cuts defined over subsets of the set partitioning constraints (3). In the general case, subset-row inequalities are expressed as:

$$\sum_{r\in\Omega} \left\lfloor \frac{1}{k} \sum_{i\in W} a_{ir} \right\rfloor x_r \leq \left\lfloor \frac{|W|}{k} \right\rfloor, \quad \forall W \subseteq \mathcal{N}_c,\ 2 \leq k \leq |W|.$$

Following Jepsen et al. (2008), we only consider cuts obtained with $|W| = 3$ and $k = 2$, resulting in

$$\sum_{r\in\Omega_W} x_r \leq 1, \quad \forall W \subseteq \mathcal{N}_c \ : |W| = 3, \tag{42}$$

where Ω_W is the subset of paths visiting at least two customers in W. In a column generation method, the addition of subset-row inequalities to the RMP requires several adjustments to the pricing problem (see, e.g., Desaulniers et al. 2011), as well as careful management when the number of added cuts increases. The implementation proposed in this study, follows the one described in Desaulniers et al. (2008). Note that, given the length of the routes generated for our tests, it was not worthwhile to apply a more sophisticated variant of the subset-row cuts such as the limited-node-memory subset-row cuts (Pecin et al. 2017b).

The second family of inequalities we consider is the 2-path cuts, introduced by Kohl et al. (1999) for the VRPTW. Let $W \subseteq \mathcal{N}_c$ be a subset of nodes such that it is not possible (e.g., because of the time windows or the vehicle capacity) to serve all customers in W by a single route. The corresponding 2-path inequality is given by

$$\sum_{r\in\Omega} \mu_r^W x_r \geq 2, \tag{43}$$

where μ_r^W is the number of times route r "enters" the set W, i.e., $\mu_r^W = \sum_{i\in\mathcal{N}\setminus W} \sum_{i\in W} \eta_{ijr}$, where non negative parameter η_{ijr} is equal to the number of times route r traverses arc $(i, j) \in \mathcal{A}$. We separate these cuts by a heuristic procedure similar to that of Desaulniers et al. (2008). In particular, we enumerate all subsets $W \subseteq \mathcal{N}_c$ such that 1) $|W|$ is less than or equal to a given value (15 in our studies), 2) the flow entering W is strictly less than two, and 3) the nodes in W are connected in the support graph of the current linear relaxation solution. If $\sum_{i\in W} q_i > Q$, then a violated inequality is found. Otherwise, we use dynamic programming to solve a minimum Hamiltonian path problem with time windows over the set $W \cup \{o, d\}$. If the problem is infeasible, a violated inequality is found. The new violated inequalities are added to the RMP. Furthermore, the dual variable associated with the 2-path cut for subset W must be subtracted from the modified arc cost $\bar{c}_{ij}$ for all arcs $(i, j) \in \mathcal{A}$ with $i \in \mathcal{N} \setminus W$ and $j \in W$.

4.4 Branching strategies

To enforce integrality, we apply the following branching scheme whenever required. First, we branch on the total number of vehicles used. If this number is integer, we branch on the arc-flow variables. In this case, we select the arc (i, j) with flow closest to 0.5. To fix the flow on this arc to 0, we simply remove (i, j) from set $\mathcal{A}$. To fix it to 1, we remove from $\mathcal{A}$ all arcs $(i, \ell), \ell \neq j$ if $i \neq 0$ and all arcs $(\ell, j), \ell \neq i$ if $j \neq n + 1$. The columns in the RMP containing any removed arc are deleted. Finally, the branch-and-bound tree is explored using a best-first strategy.

4.5 Alternative algorithm

To assess the effectiveness of the proposed BPC algorithm (denoted BPC-FC-PP because it handles the fragility constraint in the pricing problem), we implemented another BPC algorithm that does not exploit the fragility-feasible route characterization developed in Section 3. In fact, one may suspect that fragility-infeasible routes are relatively rare and should not often be encountered during the solution process. Therefore, it might not be worthwhile to consider the fragility constraint while solving the pricing problem and infeasible path cuts should rather be added to the linear relaxation whenever a linear relaxation solution contains a fragility-infeasible route. Infeasible path cuts have been used in the vehicle routing literature to handle difficult-to-model constraints pertaining to individual routes (see, e.g., Cordeau 2006; Cherkesly et al. 2016). Given an infeasible route r^* composed of a subset of arcs $\mathcal{A}_{r^*}$, the corresponding infeasible path cut writes as:

$$\sum_{r \in \Omega'} \sum_{(i,j) \in \mathcal{A}_{r^*}} \eta_{ijr} x_r \leq |\mathcal{A}_{r^*}| - 1, \tag{44}$$

where Ω' is the set of routes that are feasible with respect to the time windows and the vehicle capacity and, for each arc $(i, j) \in \mathcal{A}$, non negative parameter η_{ijr} is again equal to the number of times route $r \in \Omega'$ traverses arc (i, j).

In summary, the alternative BPC algorithm is identical to the BPC algorithm described above except that 1) the labeling algorithm does not consider the $A^{\mathcal{H}}$, $A^{\mathcal{L}}$ and X label components, nor the condition (34) in the dominance rule; and 2) infeasible path cuts of the form (44) are generated as needed whenever a linear relaxation solution contains a fragility-infeasible route. To highlight that the fragility constraint is treated in the master problem, this BPC algorithm is denoted BPC-FC-MP.

5 Computational experiments

In this section, we present the results of the computational experiments that we conducted to assess the effectiveness of the proposed BPC algorithm and to compare the solutions of the F-VRPTW obtained when varying the number of stacks and against the VRPTW solutions. In Section 5.1, we present the instances used for these tests. In Section 5.2, we describe the tests performed and which algorithms were involved in these tests. In Section 5.3, we report the computational results obtained and discuss computational performance. Finally, a comparison between the costs of the F-VRPTW and VRPTW solutions is realized in Section 5.4.

5.1 Test instances

To perform our tests, we first selected the 12 instances in the class R1 of the well-known Solomon's VRPTW benchmark instances (that can be found at http://w.cba.neu.edu/∼msolomon/r101.htm). In these instances involving 100 customers each, the customers are randomly located in a square grid and the time windows are relatively narrow with respect to the traveling times. These instances differ only by their customer time windows. Because the fragility constraint has more chances to be binding if vehicle capacity is relatively tight, i.e., if the filling rate of the vehicles is close to 100%, and the routes do not contain too many customers in $\mathcal{H}$ and $\mathcal{L}$ intertwined (see the end of Section 3), we have considered different tight vehicle capacities, namely, $Q = 48, 60, 72$. To study the impact of the number of stacks on the computational times and solution costs, we have also considered different stack heights relevant in practice, namely, $k = 2, 3, 4, 6$, yielding varying numbers of stacks $m = Q/k$ for each tested vehicle capacity Q.

Finally, for each instance, we assumed that a proportion ρ of the customers belong to $\mathcal{H}$ (with non-fragile items) and, thus, $(1 - \rho)$ belong to set $\mathcal{L}$ (with fragile items). We considered three different ρ values, namely, 25%, 50%, and 75%. To determine to which set each customer in $\mathcal{N}_c$ belongs to, we used the following procedure. Let us assume that the customers are numbered from 1 to n according to the order provided by Solomon. Then, if $\rho = 50\%$, the customers with a number equal to $2i$, $i = 1, 2, 3, \ldots$ belongs to $\mathcal{H}$; if $\rho = 25\%$, the customers with a number equal to $4i - 3$, $i = 1, 2, 3, \ldots$ belongs to $\mathcal{H}$; and, if $\rho = 75\%$, the customers with a number equal to $4i - 1$, $i = 1, 2, 3, \ldots$ belongs to $\mathcal{L}$.

Given the difficulty to solve some of these 100 customers instances to optimality, we created smaller instances by extracting the first 50 customers or the first 75 customers of each 100-customer instances. Thus, for $\rho = 50\%$, we ran tests on a total of $12 \times 3 \times 4 \times 3 = 432$ instances. For $\rho = 25\%$ and $\rho = 75\%$, we also performed tests on instances with 75 customers, $Q = 60$ and $k = 2, 3, 4, 6$.

5.2 Computational campaign and environment

Our computational experiments proceeded as follows. First, to assess the efficiency of algorithm BPC-FC-PP, we solved the 432 instances with $\rho = 50\%$ using both BPC algorithms, i.e., BPC-FC-PP and BPC-FC-MP. Then, to evaluate the impact of considering the fragility constraint on the computational times and solution costs, we also solved the corresponding VRPTW solutions using a classic BPC algorithm, i.e., algorithm BPC-FC-MP without generating any infeasible path cuts. Note that all these algorithms do not include all state-of-the-art strategies, such as route enumeration, variable fixing, or limited-memory Chvátal-Gomory rank-one inequalities (see Costa et al. 2019, for details). Therefore, the computational times achieved for the VRPTW instances are not competitive with those obtained by the state-of-the-art algorithms of Pecin et al. (2017a) and Sadykov et al. (2020).

The results collected from these experiments allow to perform sensitivity analyses with respect to the number of customers, the vehicle capacity, and the stack height, but not with respect to the proportion of customers in sets $\mathcal{L}$ and $\mathcal{H}$. For the latter, we also solved the F-VRPTW instances with $\rho = 25\%$ and $\rho = 75\%$ (for 75 customers, $Q = 60$ and $k = 2, 3, 4, 6$) using algorithm BPC-FC-PP only.

All algorithms were coded in C/C++ using the Gencol 4.5 library and the primal simplex algorithm of Cplex 12.6.3 for solving the RMPs. Our tests were performed on a PC equipped with eight Intel Core i7-4770 processors clocked at 3.40 GHz, and 16 Gb of RAM. For all runs, a single core was used and a 2-hour time limit was imposed for solving each instance.

5.3 Computational performance comparison

We start by presenting the results of our first experiment where, as mentioned above, we solved 432 instances using three algorithms. The detailed results for each individual instance can be found in Appendix B. We report in Table 2 average results computed over groups of instances with the same number of customers, vehicle capacity, and stack height. The results in this table are displayed in three blocks of columns. The first block presents for each F-VRPTW instance group, the results obtained by algorithm BPC-FC-PP, namely: the total number of instances (out of 12) solved to optimality within the time limit (#Opt); the average computational time in seconds (T); the average integrality gap computed as $(z_{IP} - z_{LP})/z_{IP}$, where z_{IP} and z_{LP} are the optimal values of the problem and the root linear relaxation, respectively (Gap); and the average number of nodes explored in the search tree (#Nodes). The second block provides the results obtained by algorithm BPC-FC-MP: the first four columns give the same information as for BPC-FC-PP; the last column indicates the average number of infeasible path cuts (44) generated (#IPCs). The third block presents for the VRPTW version of the instances, the total number of instances solved to optimality (#Opt) and the average computational time (T). Note that all averages are computed over the instances that were solved to optimality within the 2-hour time limit by the corresponding algorithm. Finally, for each number of customers ($n = 50, 75, 100$), Table 2 contains a row giving totals and averages over all instances with the same number of customers.

Table 2: Average results by number of customers, capacity and stack height ($\rho = 50\%$)

			F-VRPTW									VRPTW	
			BPC-FC-PP				BPC-FC-MP						
n	Q	k	#Opt	T (s)	Gap (%)	#Nodes	#Opt	T (s)	Gap (%)	#Nodes	#IPCs	#Opt	T (s)
50	48	2	12	2	1.02	11	12	14	1.46	166	7	12	1
		3	12	2	1.13	18	12	24	1.70	324	11	12	1
		4	12	6	1.07	43	12	237	2.05	2827	252	12	1
		6	12	3	1.08	16	12	109	1.81	1536	105	12	1
	60	2	12	15	1.42	65	12	13	1.42	78	0.2	12	13
		3	12	16	1.42	69	12	17	1.45	96	1	12	13
		4	12	15	1.39	61	12	15	1.44	82	1	12	13
		6	12	15	1.42	60	12	20	1.55	121	5	12	13
	72	2	12	11	1.43	25	12	10	1.49	35	0.9	12	9
		3	12	10	1.40	23	12	9	1.48	29	0.2	12	9
		4	12	10	1.42	24	12	11	1.49	35	0.9	12	9
		6	12	10	1.39	26	12	10	1.49	35	1	12	9
	Tot/Avg		144	9	1.29	35	144	40	1.56	447	32	144	8
75	48	2	12	106	0.71	462	12	508	0.83	4402	218	12	153
		3	12	302	0.63	1031	10	1046	0.75	7388	470	12	153
		4	12	564	0.68	1880	8	1657	0.92	10942	809	12	153
		6	12	170	0.72	574	4	1644	0.98	9556	959	12	153
	60	2	12	198	1.08	344	12	232	1.21	548	8	12	211
		3	12	220	1.08	397	12	287	1.22	747	33	12	211
		4	12	237	1.10	425	11	870	1.33	3177	140	12	211
		6	12	369	1.20	572	8	1427	1.48	4292	267	12	211
	72	2	9	945	1.40	1126	8	808	1.41	1221	19	9	497
		3	9	1279	1.43	1420	8	1403	1.50	2127	40	9	497
		4	9	1159	1.49	1415	7	1504	1.52	2872	130	9	497
		6	9	701	1.39	839	7	763	1.48	1433	55	9	497
	Tot/Avg		132	475	1.04	844	107	908	1.20	3684	220	132	268
100	48	2	11	668	0.63	2100	8	2440	0.87	14795	797	11	895
		3	10	614	0.62	1926	8	1877	0.86	9082	522	11	895
		4	11	1422	0.57	4248	5	4160	1.02	26056	2370	11	895
		6	12	2152	0.57	5025	2	3387	1.04	22433	1716	11	895
	60	2	8	1187	0.97	2366	7	2264	1.06	5669	75	8	802
		3	8	1712	1.05	2035	4	2603	1.21	9312	441	8	802
		4	8	2961	1.10	4137	2	3516	1.07	4513	187	8	802
		6	9	2260	1.01	2295	1	88	1.04	859	94	8	802
	72	2	9	1678	1.18	1639	5	2862	1.28	4146	75	10	1044
		3	10	1813	1.16	1386	4	1804	1.28	2792	51	10	1044
		4	9	1335	1.14	1156	2	645	1.27	2731	100	10	1044
		6	10	1824	1.12	1080	2	566	1.27	2674	106	10	1044
	Tot/Avg		115	1614	0.90	2506	50	2389	1.06	9956	592	116	230

To assess the efficiency of BPC-FC-PP, let us compare the results obtained by BPC-FC-PP and BPC-FC-MP. First, we observe that both algorithms can solve all 50-customer instances within the time limit but BPC-FC-PP requires significantly less time on average. As the number of customers increases, BPC-FC-PP can solve a larger proportion of instances to optimality than BPC-FC-MP. In fact, the former succeeded to solve 115 instances with 100 customers, compared to only 50 instances for the latter. Thus, we can say that BPC-FC-PP clearly outperforms BPC-FC-MP. This is obviously explained by how the fragility constraint is treated in each algorithm because this is the only difference between these two algorithms. Handling it in the pricing problem increases the difficulty of solving this pricing problem but, as the results show, it yields substantially smaller average integrality gaps (note that the gap differences would probably be much larger for the instances with 75 and 100 customers if the same number of instances were solved to optimality by both algorithms) and, therefore, less branch-and-bound nodes. Also, the algorithm BPC-FC-MP spends additional time generating infeasible path cuts. In this regard, observe that, on average, more of these cuts are generated for smaller vehicle capacities showing, as expected, that the fragility constraint is more constraining for small-capacitated vehicles. The same is observed, in general, for larger stack heights although this might be less obvious when the number of optimally-solved instances drops as k increases. Finally, when comparing the BPC-FC-PP results with those obtained for the VRPTW, we remark that about the same number of instances are solved to optimality (in total, 115 versus 116) but that the VRPTW instances are easier to solve (1614 versus 230 seconds). This is not surprising given that taking the fragility constraint into account in BPC-FC-PP increases the difficulty of solving the pricing problem.

Next, we study separately the impact of the vehicle capacity and the stack height on the computational performance of algorithm BPC-FC-PP. Table 3 reports the average results obtained by BPC-FC-PP by group of 144 instances with the same vehicle capacity (still with $\rho = 50\%$). For each instance group, the same statistics as in Table 2 are provided. These results (in particular, the total number of instances solved to optimality) indicate that the difficulty of solving the F-VRPTW increases with the vehicle capacity. Indeed, increasing vehicle capacity allows longer routes to be generated and makes the pricing problem harder to solve. Furthermore, as shown by the results, the average integrality gaps tend to increase with vehicle capacity, requiring more effort to derive an optimal integer solution. However, this is not corroborated by the average number of nodes explored that decreases when Q increases. This may be explained by the fact that, compared to the case $Q = 48$, less instances were solved to optimality and more subset-row cuts were generated for the cases $Q = 60$ and 72. Note that, from these results, it is not easy to see if the impact of handling the fragility constraint varies with the vehicle capacity. Nonetheless, given that the labeling algorithm used for the F-VRPTW requires an additional dominance condition (34) compared to the algorithm used for the VRPTW, we can speculate that handling the fragility constraint contributes to the increased difficulty of solving the pricing problem.

Table 3: Average BPC-FC-PP results by vehicle capacity ($\rho = 50\%$)

Q	#Opt	T (s)	Gap (%)	#Nodes
48	140	491	0.79	1413
60	129	622	1.21	873
72	122	826	1.32	772

Using the same format as in Table 3, Table 4 displays the average BPC-FC-PP results for groups of 108 instances with the same stack height. Here, the variation between the results obtained for the different stack heights is much less pronounced. In fact, the number of solved instances for each stack height is the same except for $k = 6$ for which three additional instances were solved. On the other hand, a slight increase of the average computational time is observed when k increases. Given that the average integrality gaps are relatively constant with respect to stack height, this small computational time increase might be due to slightly harder-to-solve pricing problems. In fact, one can observe that, for a given route r represented by label E_r, if the number of collected non-fragile items $A_r^{\mathcal{H}}$ is greater than or equal to $k - 1$, then condition ($c2$) of Proposition 1 is always fulfilled for any subsequently visited customer in $\mathcal{L}$ and X_r is often equal to $Q - L_r$. In this case, dominance condition (32) becomes redundant with condition (34) and the chances that E_r dominates another label are increased. Thus, according to this hypothesis, a larger value of k may reduce the number of dominated labels.

We conducted a second series of experiments to see if the proportion of customers in set $\mathcal{H}$ influences the computational time. In this series, we solved all 75-customer instances with $Q = 60$ and three different ρ values, namely, 25%, 50%, and 75%, using the BPC-FC-PP algorithm. Using the same statistics as above, the average results (over 48 instances per row) are reported in Table 5. These results indicate that the instances with $\rho = 50\%$ seem easier to solve than those with $\rho = 25\%$ and $\rho = 75\%$: compared to the latter, the former exhibit average time reductions of 40% and 27%,

Table 4: Average BPC-FC-PP results by stack height ($\rho = 50\%$)

k	#Opt	T (s)	Gap (%)	#Nodes
2	97	458	1.09	802
3	97	578	1.10	831
4	97	740	1.10	1362
6	100	775	1.09	1143

respectively. It should be noticed that the high average time for the instances with $\rho = 25\%$ is mostly due to one outlier (instance R104 with $k = 4$) which required more than 6700 seconds compared to less than 1800 seconds for all others. Removing this outlier would bring down the average time to around 290 seconds for the instances with $\rho = 25\%$, that is, still a 12% difference. Consequently, a balanced mix of customers with fragile and non-fragile items tends to ease the solution process, but not substantially. A possible reason for explaining this tendency is that the fragility constraint is more binding when ρ is close to 50% and, thus, reduces more the solution space as discussed at the end of the next section.

Table 5: Average BPC-FC-PP results by ρ value ($n = 75$, $Q = 60$)

ρ	#Opt	T (s)	Gap (%)	#Nodes
25%	48	425	1.03	640
50%	48	256	1.12	434
75%	48	351	1.04	578

5.4 Solution cost comparison

In this section, we compare the solutions obtained for the VRPTW and the F-VRPTW, namely, we compute the number of times that the optimal solution computed for the VRPTW is not feasible for the F-VRPTW and the induced average cost increase. As in Section 5.3, we analyze the results with respect to the number of customers, the vehicle capacity, the stack height, and the proportion of customers in $\mathcal{H}$. The results are presented in Tables 6 to 9. In all these tables, we report the following information by group of instances (the group definition depends on the table): the total number of instances for which the F-VRPTW was solved to optimality by either the BPC-FC-PP or the BPC-FC-MP algorithm (#Opt); the total number of VRPTW solutions computed for these instances that are not feasible for the F-VRPTW (#ModS) and, in parentheses, the percentage of instances with a modified solution, i.e., $100\frac{\#ModS}{\#Opt}$; and the average cost increase (in percentage) induced by the modified solutions (Δz_{IP}).

Table 6 provides the results with respect to the number of customers, i.e., each row summarizes the results for all instances with the same number of customers. From these results, we can clearly observe that the number of instances impacted by the fragility constraint increases with the number of customers, going from 44% when $n = 50$ to 89% when $n = 100$. This can be explained by the fact that the average number of routes in an optimal solution increases with the number of customers yielding more chances that at least one route in the solution is fragility-infeasible. The results also show that the average cost increase is relatively low (between 0.35% to 0.61%) and tends to decrease with the number of customers. Here, it is likely that more customers offer more options to find alternative fragility-feasible routes.

Table 6: Average cost increase by number of customers ($\rho = 50\%$)

n	#Opt	#ModS	Δz_{IP}(%)
50	144	63 (44%)	0.61
75	132	97 (73%)	0.40
100	115	102 (89%)	0.35

In Table 7, we present the results obtained when grouping the instances by vehicle capacity. From these results, we observe that the number of instances with a modified solution tends to decrease with vehicle capacity. This is to be expected because, as routes contain more customers, it becomes easier to find a feasible loading configuration. However, given that the proportion of instances with a modified solution seems to stall between $Q = 60$ (59%) and $Q = 72$ (58%), we ran additional tests with $Q = 84$. For this capacity, the proportion decreases to 38%, confirming the observed trend. On the other hand, the results also indicate that the fragility constraint impacts less the solution cost as the vehicle capacity increases. This may be explained by the fact that, when capacity increases, the solutions contain less routes and, therefore, less of them are fragility-infeasible and need to be changed.

Table 7: Average cost increase by capacity ($\rho = 50\%$)

Q	#Opt	#ModS	$\Delta z_{IP}(\%)$
48	140	115 (82%)	0.57
60	129	76 (59%)	0.38
72	122	71 (58%)	0.26

Next, we study the impact that handling the fragility constraint has on the cost of the solutions with respect to the stack height. The results for instances grouped by stack height are reported in Table 8. Without surprise, they indicate that the number of instances with a modified solution and the average cost increase are influenced by stack height. Over all instances with the same height, we can observe an increase from 55% to 78% of the #ModS statistic when k goes from 2 to 6, and an increase from 0.30% to 0.55% of Δz_{IP}. These increases are simply due to a reduction of the solution space that occurs when k increases.

Table 8: Average cost increase by stack height ($\rho = 50\%$)

k	#Opt	#ModS	$\Delta z_{IP}(\%)$
2	97	53 (55%)	0.30
3	97	61 (63%)	0.34
4	97	70 (72%)	0.48
6	100	78 (78%)	0.55

Finally, Table 9 provides results for instances grouped by proportion of customers in $\mathcal{H}$. They indicate that the fragility constraint is much more binding when there is a balance mix of customers with fragile and non-fragile items as 71% of the optimal VRPTW solutions are fragility-infeasible when $\rho = 50\%$, whereas only 23% and 33% of the optimal solutions are infeasible for $\rho = 25\%$ and $\rho = 75\%$, respectively. Furthermore, the cost increase is also larger for $\rho = 50\%$ as more routes in the VRPTW solutions become infeasible for the F-VRPTW. We explain this behavior as follows. As ρ tends toward 0%, the optimal solution routes tend to contain only customers in $\mathcal{L}$. They are, thus, fragility-feasible. Symmetrically, when ρ tends toward 100%, the routes tend to be composed of customers in $\mathcal{H}$ only and are, thus, also fragility-feasible. Therefore, the maximum number of routes that include both customer types and that are at risk of being fragility-infeasible occur when $\rho = 50\%$.

Table 9: Average cost increase by ρ value ($n = 75$, $Q = 60$)

ρ	#Opt	#ModS	$\Delta z_{IP}(\%)$
25%	48	11 (23%)	0.32
50%	48	34 (71%)	0.49
75%	48	16 (33%)	0.22

6 Conclusions

In this paper, we studied the F-VRPTW, a new variant of the VRPTW that considers a fragility constraint restricting the positioning of the items in the stacks of the vehicles. We first established necessary and sufficient conditions that characterize the feasibility of a route with respect to the fragility constraint. Then, we developed a BPC algorithm, denoted BPC-FC-PP, which exploits this characterization to handle efficiently the fragility constraint in the pricing problem. To evaluate this algorithm, we performed computational experiments on instances derived from some well-known VRPTW benchmark instances and that have been designed to yield a relatively tight fragility constraint. We compared the computational performance of BPC-FC-PP with that of another BPC algorithm, denoted BPC-FC-MP, that handles the fragility constraint in the master problem through infeasible path cuts. This comparison shows that handling the fragility constraint in the pricing problem is much more efficient than in the master problem: out of 432 instances, BPC-FC-PP solved 391 instances to optimality within a 2-hour time limit, whereas BPC-FC-MP could only solve 301 instances. Finally, we analyzed the impact that the fragility constraint has on the optimal solution cost with respect to the VRPTW solutions. Our results indicate that, for the tested instances, the computed VRPTW optimal solution is infeasible for the F-VRPTW in 67% of the cases. This proportion increases with the number of customers and the stack height, but decreases with vehicle capacity. We also observed that the fragility constraint is at its maximal tightness when there is a balance mix of customers with fragile items and with non-fragile items. For the instances requiring a modified solution, an average cost increase of around 36% is incurred.

As possible future works, we can think about tackling more general problem variants. In particular, stacks of various heights could be considered as well as fragile items and non-fragile items that do not take the same space in a stack.

Appendix

A An arc-flow formulation for the pricing problem

In this appendix, we propose an arc-flow formulation for the column generation pricing problem. For each $(i,j) \in \mathcal{A}$, we introduce a binary variable y_{ij} that takes value 1 if the route traverses arc (i,j), and 0 otherwise. For each node $i \in \mathcal{N}$, let T_i be a continuous variable indicating the service starting time. Furthermore, for each $(\gamma,\beta) \in M \times K$ potential position of an item in the vehicle and each customer $i \in \mathcal{N}_c$, we introduce a binary variable $z_{i\gamma\beta}$ which is equal to 1 if an item of customer i is placed in position (γ,β), and 0 otherwise.

The pricing problem can then be formulated as follows:

$$\text{minimize} \quad \sum_{(i,j)\in\mathcal{A}} \bar{c}_{ij} y_{ij}, \tag{45}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{N}_c} y_{oj} = 1, \tag{46}$$

$$\sum_{j\in\mathcal{N}} y_{ji} - \sum_{j\in\mathcal{N}} y_{ij} = 0, \qquad \forall i \in \mathcal{N}_c, \tag{47}$$

$$\sum_{j\in\mathcal{N}_c} y_{j,n+1} = 1, \tag{48}$$

$$T_j \geq (T_i + t_{ij}) y_{ij}, \qquad \forall (i,j) \in \mathcal{A}, \tag{49}$$

$$\underline{w}_i \leq T_i \leq \overline{w}_i, \qquad \forall i \in \mathcal{N}_c, \tag{50}$$

$$\sum_{i\in\mathcal{N}_c} z_{i\gamma\beta} \leq 1, \qquad \forall (\gamma,\beta) \in M \times K, \tag{51}$$

$$\sum_{(\gamma,\beta)\in M\times K} z_{i\gamma\beta} = q_i \sum_{j\in\mathcal{N}} y_{ij}, \qquad \forall i \in \mathcal{N}_c, \tag{52}$$

$$\sum_{i\in\mathcal{N}_c} z_{i\gamma\beta} \geq \sum_{j\in\mathcal{N}_c} z_{j\gamma(\beta+1)}, \qquad \forall (\gamma,\beta) \in M \times K \backslash \{k\}, \tag{53}$$

$$\sum_{i\in\mathcal{N}_c} T_i z_{i\gamma\beta} \leq \bar{T} + \sum_{j\in\mathcal{N}_c} (T_j - \bar{T}) z_{j\gamma(\beta+1)}, \qquad \forall (\gamma,\beta) \in M \times K \backslash \{k\}, \tag{54}$$

$$\sum_{i \in \mathcal{L}} z_{i\gamma\beta} + \sum_{j \in \mathcal{H}} z_{j\gamma(\beta+1)} \leq 1, \qquad \forall(\gamma, \beta) \in M \times K \backslash \{k\}, \tag{55}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall(i, j) \in \mathcal{A}, \tag{56}$$

$$z_{i\gamma\beta} \in \{0, 1\}, \qquad \forall i \in \mathcal{N}_c, \forall(\gamma, \beta) \in M \times K, \tag{57}$$

$$T_i \geq 0 \qquad \forall i \in \mathcal{N}, \tag{58}$$

where $\bar{T}$ is an upper bound on the service starting time at a customer and $T_0 = 0$.

Objective function (45) minimizes the total reduced cost of the route, where parameters $\bar{c}_{ij}$ are computed according to (22). Constraints (46)–(48) ensure that the y_{ij} variables describe a path in G between the origin depot node 0 and the destination depot node $n + 1$. Constraints (49) impose lower bounds on the service starting times, while (50) enforce time window restrictions. Relations (51)–(55) regulate the loading rules. In particular, inequalities (51) ensure that no more than one item is put in a given position, equalities (52) impose that the demand of each visited customer is fully satisfied, inequalities (53) imply that items are stacked on the top of others, while constraints (54) ensure coherence between the order of the customer visits and the relative positions of the items in each stack. Furthermore, relations (55) impose that no fragile item will be put under a non-fragile one. Finally, (56)–(58) define the variable domains.

Note that the capacity constraint is not expressed explicitly but rather implicitly by the number of available positions in $M \times K$.

B Detailed computational results

This appendix presents the detailed results obtained from our computational experiments. They are reported in Tables 10–20, where each table is dedicated to a number of customers (n), a capacity (Q), and a proportion of customers in $\mathcal{H}$ (ρ). For each instance and each stack height $k = 2, 3, 4, 6$, each table specifies the following information:

z_{IP}: The best solution cost found by algorithm BPC-FC-PP or BPC-FC-MP. It corresponds to the optimal value if one of these algorithms did not reach the time limit. If both algorithms reached it, an upper index indicates which algorithm (P for BPC-FC-PP and M for BPC-FC-MP) found the best cost;

#Veh: The number of vehicles in the corresponding solution;

T: The total computational time in seconds (for algorithms BPC-FC-PP and BPC-FC-MP, as well as for solving the VRPTW). TL indicates that the 7200-second time limit has been reached;

Gap: The integrality gap in percentage (for algorithms BPC-FC-PP and BPC-FC-MP) if z_{IP} is an optimal value. This gap is computed as $(z_{IP} - z_{LP})/z_{IP}$, where z_{LP} is the lower bound achieved at the root node before adding cuts;

#Nodes: The total number of nodes explored in the search tree (for algorithms BPC-FC-PP and BPC-FC-MP);

#IPCs: The total number of infeasible path cuts generated (for algorithm BPC-FC-MP);

Δz_{IP}: The difference in percentage between the optimal values of the F-VRPTW and the VRPTW, whenever z_{IP} is an optimal value. Note that all VRPTW optimal values have been computed even if some of them required to exceed the time limit.

Note that Tables 19 and 20 do not report any results for algorithm BPC-FC-MP.

Table 10: Results for $n = 50$, $Q = 48$ and $\rho = 50\%$

	F-VRPTW									VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	Δz_{IP}(%)
						$k = 2$					
R101	1255.8	17	0.1	1.1	7	0.4	1.45	41	3	0.1	0.29
R102	1166.3	17	0.6	0.86	11	0.5	0.86	11	0	0.5	0
R103	1140.5	16	2.8	1.13	29	1.5	1.13	15	0	1.6	0
R104	1116.6	16	1.8	0.99	7	29.2	1.69	309	7	2	0.51
R105	1193.8	17	0.1	0.67	1	0.7	1.19	37	5	0.2	0.37
R106	1146	16	0.4	0.81	1	2.6	1.74	47	2	1.1	0.57
R107	1126.9	16	1.6	1.18	5	18.8	1.72	219	2	2.1	0.58
R108	1116.6	16	4.8	1.01	29	99.7	1.7	1109	55	3	0.51
R109	1141.5	16	0.9	0.94	15	3.1	1.6	91	6	0.4	0.92
R110	1115	16	1.4	1.03	7	0.9	1.03	5	1	1	0
R111	1126.3	16	0.9	1.38	1	7.1	2.26	95	4	0.7	1.18
R112	1110.9	16	2.4	1.2	15	2	1.2	15	1	1.7	0
						$k = 3$					
R101	1259.6	17	0.3	1.1	17	0.9	1.75	75	9	0.1	0.59
R102	1173.5	17	1	1.18	13	5.2	1.47	117	10	0.5	0.62
R103	1142.9	16	2	1	11	5.2	1.34	69	3	1.6	0.21
R104	1116.6	16	2.6	0.99	9	33.7	1.69	372	8	2	0.51
R105	1197.8	17	0.4	0.9	13	3.5	1.52	169	5	0.2	0.71
R106	1154.9	16	0.8	1.18	5	28.6	2.5	473	31	1.1	1.35
R107	1130	16	6.4	1.4	59	64.9	1.99	762	31	2.1	0.86
R108	1116.6	16	6.9	1.01	45	100.8	1.7	1087	21	3	0.51
R109	1141.5	16	0.9	1	15	3.1	1.6	91	3	0.4	0.92
R110	1115	16	1.4	1.03	5	0.9	1.03	5	1	1	0
R111	1130.2	16	1.8	1.54	7	37	2.59	653	7	0.7	1.53
R112	1110.9	16	2.2	1.2	11	1.8	1.2	12	3	1.7	0
						$k = 4$					
R101	1255.8	17	0	0.54	1	0.3	1.45	29	5	0.1	0.29
R102	1179.5	16	0.7	0.62	7	7.6	1.97	189	6	0.5	1.13
R103	1150.1	16	3.3	1.29	33	58	1.96	991	53	1.6	0.84
R104	1124.2	16	27.6	1.28	183	565	2.36	5883	640	2	1.2
R105	1193.8	17	0.1	0.35	1	0.6	1.19	25	9	0.2	0.37
R106	1157.1	16	0.8	1.16	3	30.3	2.68	485	54	1.1	1.54
R107	1133.5	16	2	1.11	9	207.9	2.3	2191	108	2.1	1.17
R108	1121.8	16	23.6	1.32	153	1017	2.16	8677	697	3	0.98
R109	1153.7	16	4.3	1.57	93	137.3	2.64	3257	492	0.4	2
R110	1120.2	16	1.5	1.21	9	15.9	1.48	329	12	1	0.47
R111	1137.1	16	1.6	1.25	5	796	3.18	11847	948	0.7	2.15
R112	1110.9	16	3.2	1.2	19	2.2	1.2	18	4	1.7	0
						$k = 6$					
R101	1260.1	17	0.1	0.78	7	1.2	1.79	111	16	0.1	0.63
R102	1175.9	16	1.2	1.16	15	10.3	1.67	261	25	0.5	0.82
R103	1142.9	16	1.7	1	9	5	1.34	67	4	1.6	0.21
R104	1119.9	16	6.2	1.1	29	214	1.98	2615	253	2	0.81
R105	1197.8	17	0.3	0.79	7	2.7	1.52	139	16	0.2	0.71
R106	1154.9	16	0.7	1.13	1	24.8	2.5	447	51	1.1	1.35
R107	1134.8	16	10.3	1.46	73	557.3	2.41	9143	331	2.1	1.29
R108	1118.9	16	5.5	1.15	19	396.3	1.9	4127	404	3	0.72
R109	1143.5	16	0.5	0.89	5	4.9	1.77	137	12	0.4	1.1
R110	1115	16	1.5	0.98	5	0.9	1.03	5	1	1	0
R111	1130.2	16	1.4	1.28	3	93.2	2.59	1367	141	0.7	1.53
R112	1110.9	16	3.6	1.2	19	1.8	1.2	12	3	1.7	0

Table 11: Results for $n = 50$, $Q = 60$ and $\rho = 50\%$

	F-VRPTW									VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	Δz_{IP}(%)
					$k = 2$						
R101	1137.1	14	0.1	0.5	5	0	0.5	3	0	0	0
R102	1047.1	13	1.6	1.78	19	1.3	1.78	19	0	1.3	0
R103	987.8	13	0.8	0.56	1	0.6	0.56	1	0	0.6	0
R104	957.5	13	3.9	1.08	5	3.9	1.08	19	0	4	0
R105	1047.2	13	0.7	2	15	0.9	2	37	0	0.9	0
R106	1004.6	13	5.8	1.52	35	6.5	1.52	55	0	6.7	0
R107	977.8	13	85	2.18	361	60.1	2.18	357	0	62	0
R108	955.6	13	32.4	1.46	101	43.6	1.46	213	0	45.1	0
R109	996.2	13	0.5	1.36	3	0.9	1.36	13	2	0.3	0.1
R110	963.2	13	4	1.46	13	3.8	1.46	21	0	3.9	0
R111	974.5	13	18.9	1.79	103	16.2	1.79	115	0	16.7	0
R112	951.6	13	29.5	1.4	117	15.9	1.4	83	0	16.4	0
					$k = 3$						
R101	1137.1	14	0	0.47	1	0	0.5	5	1	0	0
R102	1047.1	13	1.4	1.78	15	1.4	1.78	24	1	1.3	0
R103	987.8	13	0.8	0.56	1	0.6	0.56	1	0	0.6	0
R104	960.5	13	22.5	1.35	79	45.6	1.39	231	1	4	0.31
R105	1047.2	13	0.5	1.79	10	0.7	2	27	1	0.9	0
R106	1004.6	13	6.9	1.52	39	5.5	1.52	45	1	6.7	0
R107	977.8	13	70.5	2.18	289	61	2.18	359	7	62	0
R108	955.6	13	47.6	1.46	187	44	1.46	213	0	45.1	0
R109	996.2	13	0.7	1.36	3	1	1.36	13	2	0.3	0.1
R110	963.2	13	3.2	1.45	11	4.2	1.46	23	1	3.9	0
R111	974.5	13	17.6	1.77	91	16.8	1.79	115	0	16.7	0
R112	951.6	13	25.1	1.4	97	18.5	1.4	96	1	16.4	0
					$k = 4$						
R101	1137.1	14	0.1	0.5	5	0	0.5	3	0	0	0
R102	1048.2	13	1.5	1.77	13	2.1	1.88	33	1	1.3	0.11
R103	987.8	13	0.9	0.52	1	0.6	0.56	1	0	0.6	0
R104	957.5	13	9.2	1.06	23	4.3	1.08	19	0	4	0
R105	1047.2	13	0.7	1.8	19	1	2	37	1	0.9	0
R106	1005.7	13	6.4	1.47	37	7.9	1.62	69	2	6.7	0.11
R107	977.8	13	63.9	2.18	293	65.8	2.18	355	5	62	0
R108	955.6	13	36.3	1.4	99	51	1.46	223	1	45.1	0
R109	996.2	13	0.8	1.26	3	1	1.36	13	2	0.3	0.1
R110	963.2	13	4.2	1.46	13	4.3	1.46	21	0	3.9	0
R111	974.5	13	21	1.79	117	17.8	1.79	115	3	16.7	0
R112	951.6	13	30.2	1.4	107	19.2	1.4	96	1	16.4	0
					$k = 6$						
R101	1143.9	14	0.1	0.85	9	0.3	1.09	43	4	0	0.6
R102	1049.7	13	2.8	1.91	33	3.9	2.02	69	5	1.3	0.25
R103	987.8	13	0.6	0.52	1	0.6	0.56	1	0	0.6	0
R104	960.5	13	21.8	1.35	73	45.7	1.39	231	1	4	0.31
R105	1048.2	13	0.5	1.61	13	1.3	2.1	45	5	0.9	0.1
R106	1005.7	13	4	1.47	23	7.4	1.62	71	3	6.7	0.11
R107	978.7	13	40	2.19	145	93	2.27	515	37	62	0.09
R108	955.6	13	55.7	1.46	203	43.8	1.46	213	0	45.1	0
R109	996.6	13	0.6	1.35	1	1.6	1.4	31	3	0.3	0.14
R110	963.2	13	3.6	1.39	13	4	1.46	23	1	3.9	0
R111	974.5	13	18.3	1.56	83	16.1	1.79	115	3	16.7	0
R112	951.6	13	32.7	1.4	117	17.6	1.4	96	1	16.4	0

Table 12: Results for $n = 50$, $Q = 72$ and $\rho = 50\%$

	F-VRPTW									VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	$\Delta z_{IP}(\%)$
					$k = 2$						
R101	1077	13	0	0.11	1	0	0.11	1	0	0	0
R102	959.8	12	0.2	0.36	1	0.5	0.69	9	2	0.1	0.23
R103	902.4	11	3.3	1.2	5	8.6	1.61	35	5	3	0.2
R104	856	11	27.2	2.03	47	19	2.03	45	0	19.6	0
R105	959.3	11	0.2	0.6	3	0.1	0.6	3	0	0.1	0
R106	905.1	11	18.6	2.46	111	16.4	2.49	135	1	11.9	0.07
R107	876.5	11	8	1.86	15	7.5	1.87	21	0	7.7	0
R108	848.8	11	43.6	2	57	42.1	2	87	0	43.1	0
R109	903.4	11	4.3	2.04	25	3.6	2.04	27	0	3.7	0
R110	848	11	0.7	0.49	1	0.4	0.49	1	0	0.4	0
R111	876.8	11	10.6	2.07	25	8.8	2.07	35	0	9	0
R112	844.6	11	8.8	1.9	5	9.7	1.9	22	3	9.2	0
					$k = 3$						
R101	1077	13	0	0.11	1	0	0.11	1	0	0	0
R102	959.2	12	0.2	0.33	1	0.2	0.63	3	1	0.1	0.17
R103	901.8	11	2.5	1.13	1	5	1.55	21	1	3	0.13
R104	856	11	23.5	2.03	37	19.2	2.03	45	0	19.6	0
R105	959.3	11	0.1	0.46	1	0.1	0.6	3	0	0.1	0
R106	904.5	11	11.5	2.43	65	11.7	2.43	81	0	11.9	0
R107	876.5	11	8.7	1.86	21	7.5	1.87	21	0	7.7	0
R108	848.8	11	37.3	2	59	42.4	2	87	0	43.1	0
R109	903.4	11	5.6	2.04	35	3.6	2.04	27	0	3.7	0
R110	848	11	0.7	0.49	1	0.4	0.49	1	0	0.4	0
R111	876.8	11	10.7	2.07	29	8.8	2.07	35	0	9	0
R112	844.6	11	14.3	1.9	23	9	1.9	19	0	9.2	0
					$k = 4$						
R101	1077	13	0	0.11	1	0	0.11	1	0	0	0
R102	959.8	12	0.3	0.36	1	0.5	0.69	9	2	0.1	0.23
R103	902.4	11	2.3	1.16	1	8.9	1.61	35	5	3	0.2
R104	856	11	24.1	2.03	37	20.2	2.03	45	0	19.6	0
R105	959.3	11	0.2	0.6	3	0.1	0.6	3	0	0.1	0
R106	905.1	11	18.2	2.46	111	17.4	2.49	135	1	11.9	0.07
R107	876.5	11	5.7	1.86	9	7.9	1.87	21	0	7.7	0
R108	848.8	11	37.2	2	57	45.3	2	87	0	43.1	0
R109	903.4	11	5.2	2.04	25	4.2	2.04	27	0	3.7	0
R110	848	11	0.9	0.49	1	0.4	0.49	1	0	0.4	0
R111	876.8	11	10.6	2.07	29	9.6	2.07	35	0	9	0
R112	844.6	11	8.9	1.9	9	11.1	1.9	22	3	9.2	0
					$k = 6$						
R101	1077	13	0	0.11	1	0	0.11	1	0	0	0
R102	959.8	12	0.2	0.36	1	0.5	0.69	9	2	0.1	0.23
R103	902.4	11	3.3	1.16	3	8.5	1.61	35	5	3	0.2
R104	856	11	28.9	2.03	39	19.6	2.03	45	0	19.6	0
R105	959.3	11	0.1	0.46	1	0.1	0.6	3	0	0.1	0
R106	905.1	11	22.4	2.46	125	16.6	2.49	135	1	11.9	0.07
R107	876.5	11	7.2	1.76	13	7.7	1.87	21	0	7.7	0
R108	848.8	11	32.8	2	47	42.7	2	87	0	43.1	0
R109	903.4	11	7.1	2.03	43	3.6	2.04	27	0	3.7	0
R110	848	11	0.8	0.38	1	0.4	0.49	1	0	0.4	0
R111	876.8	11	11.1	2.07	27	8.8	2.07	35	0	9	0
R112	844.6	11	9.4	1.9	7	9.9	1.9	22	8	9.2	0

Table 13: Results for $n = 75$, $Q = 48$ and $\rho = 50\%$

			F-VRPTW								VRPTW	
			BPC-FC-PP			BPC-FC-MP						
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	Δz_{IP}(%)
					$k = 2$						
R101	1763.3	25	1.5	0.81	52	1.7	0.87	63	5	1	0.17
R102	1694.9	24	5.7	0.68	49	8.5	0.75	77	10	6.9	0
R103	1641.9	24	23	0.58	155	82.3	0.75	678	12	74.9	0
R104	1618	24	279.2	0.75	1031	339	0.76	1573	8	318.9	0
R105	1690.6	24	1.3	0.38	13	17.4	0.67	305	36	0.3	0.46
R106	1665.5	24	104.7	1.1	809	1833.3	1.27	17113	745	34.3	0.43
R107	1622.4	24	24.7	0.59	115	51.7	0.59	297	27	27.1	0.02
R108	1617.4	24	242	0.79	813	271.4	0.79	1073	98	266.7	0
R109	1645.1	24	25.2	0.89	221	1852.5	1.23	20353	1358	18.6	0.49
R110	1622.6	24	28.3	0.6	141	960	0.92	7923	274	244.9	0.21
R111	1624.7	24	54.9	0.64	303	50	0.67	297	10	30.7	0.02
R112	1612	24	480.4	0.67	1839	630.8	0.67	3075	33	809.3	0
					$k = 3$						
R101	1760.3	24	0.7	0.63	21	0.7	0.7	27	4	1	0
R102	1702.6	24	3.7	0.54	23	196.1	1.2	1791	88	6.9	0.45
R103	1643	24	33.8	0.65	209	153.4	0.82	1249	92	74.9	0.07
R104	1621.1	24	1940.8	0.81	6247	TL	0.95	37252	2137	318.9	0.19
R105	1682.9	24	0.1	0.08	1	0.3	0.22	4	2	0.3	0
R106	1671.4	24	117.1	1	851	TL	1.62	69919	2772	34.3	0.79
R107	1622.4	24	12.5	0.5	41	44.3	0.59	263	21	27.1	0.02
R108	1619.8	24	1067.5	0.8	3193	3759.1	0.94	17115	2029	266.7	0.15
R109	1638	24	4.2	0.53	21	37.4	0.8	391	48	18.6	0.06
R110	1622.7	24	104.9	0.76	543	5605.7	0.92	49659	2241	244.9	0.22
R111	1624.7	24	16.3	0.54	73	54.1	0.67	343	27	30.7	0.02
R112	1612	24	321.4	0.67	1149	608.8	0.67	3039	149	809.3	0
					$k = 4$						
R101	1763.3	25	0.2	0.54	5	1.6	0.87	63	18	1	0.17
R102	1709.9	24	16.3	0.76	121	2979.3	1.63	30933	2852	6.9	0.89
R103	1643	24	3.1	0.44	5	134.8	0.82	1125	130	74.9	0.07
R104	1621.1	24	1170	0.76	3947	6595.5	0.95	33445	2035	318.9	0.19
R105	1696.6	24	4.8	0.59	75	231.1	1.02	4415	374	0.3	0.81
R106	1671.2	24	61.4	0.99	323	TL	1.61	68179	5060	34.3	0.78
R107	1622.4	24	9.2	0.45	25	46.1	0.59	265	32	27.1	0.02
R108	1621.1	24	2828.7	0.83	8807	TL	1.02	35024	2624	266.7	0.23
R109	1648.9	24	16.5	0.77	117	TL	1.45	79603	7937	18.6	0.73
R110	1629.7	24	130.6	0.78	661	TL	1.35	55775	3432	244.9	0.65
R111	1626.3	24	25.3	0.57	89	194.2	0.77	1347	174	30.7	0.12
R112	1613.3	24	2506.8	0.71	8387	3072.9	0.75	15941	857	809.3	0.08
					$k = 6$						
R101	1768	25	0.1	0.4	3	7.1	1.13	247	61	1	0.44
R102	1720.4	24	19.5	0.63	139	TL	2.23	59118	7965	6.9	1.5
R103	1649	24	50.3	0.73	257	2524.9	1.18	19849	1468	74.9	0.43
R104	1621.8	24	906.7	0.83	2513	TL	0.99	33415	6437	318.9	0.23
R105	1706.6	24	7.3	0.76	91	TL	1.6	106029	12788	0.3	1.41
R106	1676.9	24	86.8	0.95	562	TL	1.94	61363	6200	34.3	1.12
R107	1635.4	24	150.3	0.8	713	TL	1.38	44583	3645	27.1	0.83
R108	1619.8	24	316.8	0.78	721	3725	0.94	16497	2209	266.7	0.15
R109	1649.8	24	6.6	0.66	39	TL	1.51	80032	8339	18.6	0.78
R110	1628.5	24	18.3	0.58	75	TL	1.28	58668	4502	244.9	0.57
R111	1638.9	24	246.2	0.83	1133	TL	1.53	46670	4393	30.7	0.9
R112	1612	24	225.6	0.67	643	318.4	0.67	1638	99	809.3	0

Table 14: Results for $n = 75$, $Q = 60$ and $\rho = 50\%$

			F-VRPTW								VRPTW	
			BPC-FC-PP			BPC-FC-MP						
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs		T (s)	$\Delta z_{IP}(\%)$
						$k = 2$						
R101	1569.5	20	4.5	1.14	83	7.9	1.79	175	1		0.7	1.1
R102	1481.3	19	25	0.96	115	50	1.58	265	2		8.2	0.7
R103	1384.3	19	218.5	1.35	441	252	1.39	615	8		170.2	0.04
R104	1350.2	19	733.3	1.04	1171	1094.2	1.04	2115	3		915	0
R105	1455.4	19	1.8	0.53	8	1.8	0.55	15	1		2.1	0.01
R106	1412.1	19	14.4	1.13	25	16.8	1.15	53	1		16.3	0
R107	1359.4	19	69	1.09	109	52.5	1.14	95	6		20.9	0.15
R108	1347.1	19	352.8	1.09	425	321.4	1.09	427	6		370.9	0
R109	1382.9	19	92	1.27	353	199.9	1.31	937	13		42.1	0.2
R110	1357.1	19	41.6	1.03	67	98	1.14	295	19		67.2	0.03
R111	1365.7	19	184.2	1.36	299	211.3	1.37	477	14		228.1	0
R112	1339.6	19	635.9	1	1033	478	1	1103	18		693.6	0
						$k = 3$						
R101	1569.5	20	3.9	1.2	69	7.6	1.79	175	1		0.7	1.1
R102	1483.2	19	110.8	1.31	384	170.4	1.71	943	33		8.2	0.83
R103	1384.2	19	124.4	1.32	225	227.4	1.38	631	32		170.2	0.04
R104	1350.2	19	748.6	1.04	1123	1010	1.04	2075	30		915	0
R105	1455.4	19	1.7	0.46	5	1.6	0.55	11	2		2.1	0.01
R106	1412.1	19	19.7	1.14	35	15.9	1.15	53	1		16.3	0
R107	1359.4	19	31.5	1	37	55.8	1.14	106	11		20.9	0.15
R108	1347.1	19	410.9	1.08	463	308.8	1.09	427	6		370.9	0
R109	1382	19	109.1	1.19	339	109	1.25	563	18		42.1	0.14
R110	1357.2	19	22.9	0.87	35	77.3	1.15	235	8		67.2	0.04
R111	1365.7	19	106.5	1.33	171	200.9	1.37	435	32		228.1	0
R112	1340.4	19	945.3	0.99	1873	1253.3	1.06	3315	221		693.6	0.06
						$k = 4$						
R101	1569.6	20	1.4	0.94	21	5.7	1.8	125	2		0.7	1.1
R102	1481.3	19	14.7	0.85	39	42.2	1.58	237	7		8.2	0.7
R103	1387.3	19	318	1.47	641	1392.4	1.6	4211	194		170.2	0.26
R104	1350.2	19	840.4	1.04	1259	851.6	1.04	1753	23		915	0
R105	1462.7	19	9.4	0.92	57	42.9	1.04	457	33		2.1	0.51
R106	1412.1	19	15.5	1.05	33	16.1	1.15	53	1		16.3	0
R107	1359.4	19	25.6	0.98	21	45.5	1.14	89	16		20.9	0.15
R108	1347.1	19	267	1.08	269	290.8	1.09	411	38		370.9	0
R109	1389.4	19	83.1	1.32	239	4667	1.77	21415	1116		42.1	0.67
R110	1363	19	560.2	1.27	1347	TL	1.57	23754	1066		67.2	0.46
R111	1365.7	19	71.8	1.29	97	189.7	1.37	481	23		228.1	0
R112	1340.4	19	633.2	0.99	1071	2021.8	1.06	5711	85		693.6	0.06
						$k = 6$						
R101	1573.1	20	3.5	0.96	69	41.6	2.02	857	58		0.7	1.33
R102	1493.2	19	15.3	0.98	46	TL	2.36	36122	1971		8.2	1.51
R103	1399	19	1614.3	1.81	2693	TL	2.42	19908	1395		170.2	1.11
R104	1350.2	19	513.8	1.04	649	791.6	1.04	1581	26		915	0
R105	1467.6	19	14.1	1.1	97	373.1	1.37	3517	393		2.1	0.85
R106	1421.8	19	82.2	1.37	215	1045.3	1.82	4195	200		16.3	0.69
R107	1368.5	19	103.8	1.25	145	5631.5	1.8	14607	922		20.9	0.82
R108	1348.6	19	774.7	1.19	821	1595.7	1.2	3245	145		370.9	0.11
R109	1385.5	19	19.8	1.02	47	618.6	1.5	3279	225		42.1	0.39
R110	1363.1	19	311	1.19	679	TL	1.58	26035	2252		67.2	0.47
R111	1375.1	19	520.7	1.49	747	TL	2.04	19197	1387		228.1	0.69
R112	1340.4	19	459.8	0.99	651	1316.7	1.06	3053	167		693.6	0.06

Table 15: Results for $n = 75$, $Q = 72$ and $\rho = 50\%$

	F-VRPTW									VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	Δz_{IP}(%)
					$k = 2$						
R101	1486.2	19	0.2	0.63	7	0.3	0.68	13	1	0.3	0
R102	1367.6	17	3	1.15	9	2.1	1.22	7	0	2	0
R103	1245.9	16	4769.6	2.09	5945	TL	2.19	9154	151	442.2	0.31
R104	1198^{PM}	16	TL	-	6350	TL	-	8909	162	TL	-
R105	1329.5	16	59.6	1.75	259	175.5	1.76	821	19	55.7	0.2
R106	1271.3	16	119.4	1.59	241	198.3	1.66	525	13	195.5	0
R107	1216.8	16	890.1	1.53	737	2774.8	1.79	3399	47	1634.8	0.06
R108	1592.3^{PM}	24	TL	-	4523	TL	-	7795	54	TL	-
R109	1239.4	16	28.9	0.93	47	33.2	0.95	83	1	40.3	0
R110	1211.2	16	1874	1.63	2101	1710.8	1.64	2817	32	1648.8	0
R111	1216.5	16	760.6	1.33	791	1572.7	1.55	2099	42	450.6	0.17
R112	1672.2^{P}	26	TL	-	5148	TL	-	9769	64	TL	-
					$k = 3$						
R101	1486.2	19	0.2	0.68	7	0.3	0.68	13	0	0.3	0
R102	1370.3	17	6.7	1.34	19	9.3	1.41	59	2	2	0.2
R103	1245.9	16	3771.7	2.09	4177	6732.1	2.19	10111	266	442.2	0.31
R104	1800.1^{PM}	27	TL	-	5271	TL	-	8459	221	TL	-
R105	1329.4	16	86.4	1.73	365	179.8	1.75	859	8	55.7	0.19
R106	1271.3	16	123.9	1.52	267	153.1	1.66	447	7	195.5	0
R107	1216.8	16	1033.3	1.52	925	2644.2	1.79	3399	28	1634.8	0.06
R108	1592.3^{PM}	24	TL	-	5486	TL	-	8101	39	TL	-
R109	1239.4	16	30.4	0.87	49	32.3	0.95	83	1	40.3	0
R110	1213.3	16	5709.7	1.79	6149	TL	1.81	12878	282	1648.8	0.17
R111	1216.5	16	745.4	1.33	825	1476.4	1.55	2043	10	450.6	0.17
R112	1673^{PM}	26	TL	-	6505	TL	-	9846	94	TL	-
					$k = 4$						
R101	1486.2	19	0.2	0.63	7	0.2	0.68	13	1	0.3	0
R102	1367.6	17	3.5	1.06	9	2	1.22	7	0	2	0
R103	1245.9	16	3956	2.09	4283	TL	2.19	10372	230	442.2	0.31
R104	1200.8^{P}	16	TL	-	6491	TL	-	9704	298	TL	-
R105	1333.3	16	273.1	2.02	1175	1133.4	2.04	4237	132	55.7	0.48
R106	1276	16	345.2	1.73	803	2732.3	2.02	6673	479	195.5	0.37
R107	1217.3	16	590.9	1.52	544	3891	1.83	4908	176	1634.8	0.1
R108	1592.3^{PM}	24	TL	-	5397	TL	-	8301	89	TL	-
R109	1243.5	16	93.2	1.16	169	313.2	1.28	719	27	40.3	0.33
R110	1215.3	16	4301.9	1.8	4763	TL	1.97	11331	514	1648.8	0.34
R111	1217.2	16	868	1.38	983	2454.4	1.61	3541	95	450.6	0.23
R112	1653.8^{P}	26	TL	-	6552	TL	-	9974	112	TL	-
					$k = 6$						
R101	1486.2	19	0.2	0.62	7	0.3	0.68	13	1	0.3	0
R102	1372.8	17	16.4	1.39	61	54.1	1.59	321	24	2	0.38
R103	1245.9	16	2052.6	2.09	2257	TL	2.19	10044	426	442.2	0.31
R104	1792.7^{P}	27	TL	-	5266	TL	-	9462	315	TL	-
R105	1332.2	16	61.7	1.7	303	647	1.96	2579	97	55.7	0.4
R106	1273.3	16	105.6	1.59	207	513.8	1.81	1523	95	195.5	0.16
R107	1216.8	16	461.1	1.42	467	2580.1	1.79	3409	108	1634.8	0.06
R108	1592.3^{PM}	24	TL	-	4762	TL	-	7893	136	TL	-
R109	1239.4	16	25.4	0.67	44	32.4	0.95	81	3	40.3	0
R110	1213.3	16	3131.9	1.71	3639	TL	1.81	12855	448	1648.8	0.17
R111	1216.5	16	456.8	1.28	562	1513.2	1.55	2107	55	450.6	0.17
R112	1622.8^{P}	25	TL	-	5503	TL	-	10167	419	TL	-

Table 16: Results for $n = 100$, $Q = 48$ and $\rho = 50\%$

	F-VRPTW									VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	Δz_{IP}(%)
					$k = 2$						
R101	2165.6	33	2.1	0.49	29	11.7	0.87	245	19	4.8	0.24
R102	2105.4	32	260.7	0.83	1371	3876	1.29	25523	1381	102.2	0.49
R103	2029.2	32	55.4	0.69	133	5754.9	0.96	25957	1439	4952.8	0.02
R104	1992.5	32	809.7	0.58	1745	849.9	0.58	2688	130	487.3	0
R105	2073.8	32	9.2	0.36	53	813.2	1.04	10389	581	46.9	0.44
R106	2054.1	32	974.9	0.71	4605	TL	1.17	34409	2069	823.5	0.54
R107	2012.1	32	139	0.75	279	1304.3	0.84	4769	236	879	0
R108	1990.3	32	1250.7	0.54	3155	675.2	0.54	2165	69	888.6	0
R109	2016.7	32	148.8	0.6	885	6236.4	0.81	46627	2523	366.6	0.21
R110	1997	32	2065.2	0.65	5835	TL	0.88	33829	929	902.8	0.26
R111	2007	32	1629.3	0.7	5015	TL	0.77	29778	2492	387	0.15
R112	2740.4^{PM}	49	TL	-	21941	TL	-	35995	741	TL	-
					$k = 3$						
R101	2167.5	33	3.5	0.58	67	29.6	0.96	599	39	4.8	0.33
R102	2104.9	32	119.8	0.73	577	2450.7	1.27	14321	1199	102.2	0.47
R103	2029.2	32	205	0.78	539	6559.6	0.96	27305	1738	4952.8	0.02
R104	1995.2^{P}	32	TL	-	19935	TL	-	25430	495	487.3	-
R105	2072.3	32	29.2	0.44	273	608.8	0.97	7638	369	46.9	0.37
R106	2057.5	32	1682.4	0.62	6791	TL	1.33	2079	89639	823.5	0.71
R107	2012.1	32	226.5	0.75	465	935	0.84	2959	130	879	0
R108	1990.3	32	709	0.52	1665	602.5	0.54	1645	58	888.6	0
R109	2013.4	32	19.5	0.44	81	363.8	0.65	2521	104	366.6	0.04
R110	1993.2	32	157.8	0.51	453	3467.1	0.69	15669	535	902.8	0.07
R111	2009.7	32	2985.3	0.82	8351	TL	0.9	29208	1433	387	0.29
R112	1985.3^{P}	32	TL	-	20274	TL	-	33607	722	TL	-
					$k = 4$						
R101	2167.7	33	0.6	0.29	7	29.6	0.97	491	52	4.8	0.34
R102	2108.8	32	68.6	0.6	314	6035.9	1.45	35069	3517	102.2	0.65
R103	2034.2	32	67.4	0.61	189	TL	1.2	28476	1587	4952.8	0.27
R104	1995.2	32	5697.4	0.64	15761	TL	0.72	24463	940	487.3	0.14
R105	2077.5	32	3	0.32	27	6398.1	1.22	66163	6111	46.9	0.62
R106	2057.5	32	276.8	0.54	1203	TL	1.33	29211	1898	823.5	0.71
R107	2012.1	32	72.7	0.65	111	1281.2	0.84	4095	249	879	0
R108	1991.6	32	4096.7	0.55	11113	7054.7	0.6	24463	1923	888.6	0.07
R109	2018.6	32	48.8	0.56	253	TL	0.91	45429	4234	366.6	0.3
R110	2000.1	32	4635.1	0.75	15893	TL	1.03	27606	1627	902.8	0.42
R111	2011.4	32	671.3	0.76	1859	TL	0.98	25475	3090	387	0.37
R112	1986.6^{P}	32	TL	-	22655	TL	-	28974	1007	TL	-
					$k = 6$						
R101	2179.2	33	0.7	0.33	9	1476	1.49	25029	1788	4.8	0.87
R102	2113.3	33	120.1	0.53	535	TL	1.66	43753	5785	102.2	0.87
R103	2033.2	32	192.8	0.61	373	TL	1.15	31607	3146	4952.8	0.22
R104	1995.2	32	4094.6	0.67	7323	TL	0.72	25955	1378	487.3	0.14
R105	2087.2	32	9.3	0.35	67	TL	1.68	83450	12000	46.9	1.09
R106	2058	32	90.4	0.45	297	TL	1.36	30622	3201	823.5	0.73
R107	2018.8	32	7165.4	0.69	16709	TL	1.17	25827	2336	879	0.33
R108	1991.6	32	1104.5	0.58	1493	5297.6	0.6	19837	1643	888.6	0.07
R109	2022.9	32	157.6	0.68	751	TL	1.12	47850	3989	366.6	0.52
R110	1999.5	32	1747.5	0.63	5035	TL	1	34434	1696	902.8	0.39
R111	2015	32	4942.4	0.65	13569	TL	1.16	28075	2633	387	0.55
R112	1984.9	32	6196.2	0.65	14139	TL	0.65	33170	1502	TL	0

Table 17: Results for $n = 100$, $Q = 60$ and $\rho = 50\%$

	F-VRPTW									VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	Δz_{IP}(%)
$k = 2$											
R101	1872.8	26	18.3	0.85	165	40	0.9	445	7	35.4	0.02
R102	1799.3	25	664.1	1.1	2013	4511.2	1.5	15943	111	2601	0.17
R103	1702.5	25	2934.6	1.12	3803	3361.6	1.16	5601	78	986.6	0.11
R104	2583.7^{PM}	43	TL	-	6667	TL	-	9211	78	TL	-
R105	1774.1	25	58.5	0.97	221	129.3	1.04	617	42	61	0.08
R106	1733.8	26	296.6	0.71	539	2372.1	0.76	4497	130	641.8	0.09
R107	2688.9^{PM}	44	TL	-	6301	TL	-	9045	14	TL	-
R108	2424.4^{P}	41	TL	-	6221	TL	-	9875	586	TL	-
R109	1693.6	26	4367.1	1.06	10923	5193.4	1.12	14380	154	1261.6	0.1
R110	1668.7	25	901.8	1.05	1039	TL	1.48	14130	148	581.7	0.47
R111	1676.3	25	256.9	0.94	227	242	0.97	302	6	250.7	0
R112	2423^{PM}	43	TL	-	6205	TL	-	10762	67	TL	-
$k = 3$											
R101	1872.8	26	21.3	0.79	188	42.4	0.9	453	14	35.4	0.02
R102	1797.5	25	500.9	1.21	1211	1741.8	1.41	5367	113	2601	0.07
R103	1704.6	26	1042.3	1.08	1127	TL	1.28	11393	413	986.6	0.23
R104	1660.9^{P}	25	TL	-	6174	TL	-	8432	200	TL	-
R105	1780.7	25	309.4	1.22	1105	6946.4	1.41	29123	1568	61	0.46
R106	1737.4	25	1429.6	0.79	2265	TL	0.97	14541	370	641.8	0.29
R107	1682.6	25	6589.6	1.16	6063	TL	1.27	8054	134	TL	0
R108	2353.7^{P}	39	TL	-	6266	TL	-	8936	323	TL	-
R109	1697.9^{P}	26	TL	-	19036	TL	-	19687	370	1261.6	-
R110	1671.9	25	3268.5	1.11	3773	TL	1.67	11926	420	581.7	0.66
R111	1679.2	25	537	1.04	549	1681.2	1.14	2305	68	250.7	0.17
R112	2423^{PM}	43	TL	-	7063	TL	-	9393	305	TL	-
$k = 4$											
R101	1872.9	26	7.7	0.79	75	30.3	0.9	319	9	35.4	0.03
R102	1800.6	25	344.5	1.08	987	TL	1.58	20733	612	2601	0.24
R103	1708.9	26	6941.7	1.28	8055	TL	1.53	9963	576	986.6	0.48
R104	1660.9	25	5734.1	1.05	4579	TL	1.17	7419	430	TL	0
R105	1784.7	26	532.2	1.4	2257	TL	1.63	28087	2537	61	0.68
R106	1742	25	4719.7	0.96	6780	TL	1.23	8357	24538	641.8	0.56
R107	2688.9^{M}	44	TL	-	7087	TL	-	7986	175	TL	-
R108	2353.7^{P}	39	TL	-	6886	TL	-	8825	414	TL	-
R109	1702	26	4676.9	1.18	9649	TL	1.61	18712	854	1261.6	0.6
R110	2468.3^{P}	42	TL	-	10845	TL	-	12785	662	581.7	-
R111	1680.9	25	729.5	1.08	717	7001.1	1.24	8707	364	250.7	0.27
R112	2422.8^{P}	43	TL	-	7020	TL	-	9314	403	TL	-
$k = 6$											
R101	1875.4	26	7	0.71	73	87.6	1.04	859	94	35.4	0.16
R102	1801.9	25	240.1	1.11	563	TL	1.65	23395	611	2601	0.31
R103	1711.7	26	5820.3	1.12	5757	TL	1.69	12558	706	986.6	0.65
R104	1660.9	25	6988.9	1	4649	TL	1.17	9331	278	TL	0
R105	1787.1	25	401.2	1.28	1639	TL	1.76	15839	48926	61	0.82
R106	1746.3	25	4027.5	0.94	5305	TL	1.47	14475	788	641.8	0.81
R107	1686.3	25	1013.2	1.05	669	TL	1.48	8409	331	TL	0.22
R108	2331.4^{P}	37	TL	-	4621	TL	-	9785	312	TL	-
R109	1700.9^{P}	26	TL	-	15253	TL	-	20153	801	1261.6	-
R110	1672	25	1287.3	1.01	1499	TL	1.68	11287	1556	581.7	0.67
R111	1682.7	25	558.1	0.9	503	TL	1.34	9015	569	250.7	0.38
R112	2420.6^{P}	43	TL	-	6053	TL	-	9174	968	TL	-

Table 18: Results for $n = 100$, $Q = 72$ and $\rho = 50\%$

			F-VRPTW							VRPTW	
			BPC-FC-PP			BPC-FC-MP					
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	Gap (%)	#Nodes	#IPCs	T (s)	$\Delta z_{IP}(\%)$
					$k = 2$						
R101	1760.8	23	80.5	1.21	592	595.6	1.23	4457	105	334.9	0.06
R102	1632.2	22	371.1	1.15	603	613.5	1.3	1105	31	22.5	0.31
R103	1508.8	21	675.5	1	591	2603.4	1.32	3281	23	273.3	0.23
R104	1458.6	21	1447.5	0.98	593	TL	1.14	5476	119	1243.7	0.12
R105	1598	21	1827.6	1.38	3693	TL	1.56	15100	149	113.4	0.54
R106	1544.9	21	1715.6	1.17	1637	6584.8	1.23	7810	184	201.2	0.32
R107	2569.2^{PM}	43	TL	-	3336	TL	-	5241	23	848.5	-
R108	2271^{M}	37	TL	-	2326	TL	-	4030	33	TL	-
R109	1510	21	3126.7	1.27	3227	TL	1.37	8126	82	942.9	0.25
R110	1470.6	21	3088.5	1.23	2217	3911.2	1.31	4077	32	4729.7	0
R111	1481.1	21	2772.3	1.25	1597	TL	1.42	4818	145	1732.5	0.22
R112	2060.5^{PM}	35	TL	-	2799	TL	-	4767	112	TL	-
					$k = 3$						
R101	1760	23	62.4	1.19	451	393.2	1.19	3195	29	334.9	0.01
R102	1632.2	22	490.2	1.12	661	625.7	1.3	1115	31	22.5	0.31
R103	1508.8	21	673.5	1.03	492	2559.7	1.32	3269	76	273.3	0.23
R104	1459.1	21	1304.8	1.01	527	TL	1.17	5972	205	1243.7	0.15
R105	1597.4	22	661.1	1.22	1485	TL	1.53	14624	341	113.4	0.5
R106	1546.3	22	2146.8	1.21	2205	TL	1.32	8485	454	201.2	0.42
R107	1484.9	21	5966.8	1.2	2895	TL	1.63	5259	67	848.5	0.49
R108	2305.3^{P}	38	TL	-	2363	TL	-	3840	69	TL	-
R109	1510	21	3996.5	1.28	3417	TL	1.37	7936	228	942.9	0.25
R110	1470.6	21	584.8	1.15	383	3635.4	1.31	3589	69	4729.7	0
R111	1481.4	21	2245.2	1.22	1341	TL	1.44	4460	147	1732.5	0.24
R112	2060.5^{PM}	35	TL	-	3000	TL	-	4718	195	TL	-
					$k = 4$						
R101	1760.9	22	48.9	1.21	363	724.8	1.24	4607	152	334.9	0.06
R102	1632.2	22	141.3	1.05	193	564.8	1.3	855	48	22.5	0.31
R103	1511	21	2075	1.04	1779	TL	1.47	7667	111	273.3	0.37
R104	1459.6	21	1769	1.02	699	TL	1.2	4637	400	1243.7	0.19
R105	1600.1	21	843.5	1.23	1715	TL	1.69	13603	216	113.4	0.67
R106	1548.8	21	2928.4	1.22	2689	TL	1.48	7135	221	201.2	0.58
R107	2574.4^{P}	43	TL	-	3933	TL	-	4635	42	848.5	-
R108	1454.9^{P}	21	TL	-	2687	TL	-	3930	229	TL	-
R109	1510.2	21	1014	1.07	941	TL	1.38	7842	136	942.9	0.26
R110	1471.5	21	692.8	1.13	517	TL	1.37	7089	134	4729.7	0.06
R111	1484	21	2499.8	1.25	1511	TL	1.61	4473	186	1732.5	0.41
R112	2060.5^{PM}	35	TL	-	3885	TL	-	4209	120	TL	-
					$k = 6$						
R101	1760.8	23	51.1	1.2	407	632	1.23	4477	172	334.9	0.06
R102	1632.2	22	242.5	1.06	321	500.4	1.3	871	40	22.5	0.31
R103	1512.3	21	3884.2	1.08	2677	TL	1.55	8387	411	273.3	0.46
R104	1459.1	21	1742.1	1	511	TL	1.17	5247	433	1243.7	0.15
R105	1600.1	21	313.8	1.22	653	TL	1.69	13804	363	113.4	0.67
R106	1549.2	21	1802.5	1.07	1767	TL	1.51	8334	380	201.2	0.6
R107	1486.3	21	7062.2	1.16	2779	TL	1.72	4589	127	848.5	0.59
R108	2305.3^{P}	38	TL	-	1873	TL	-	3507	496	TL	-
R109	1510.2	21	278.3	1.02	237	TL	1.38	8284	231	942.9	0.26
R110	1471.9	21	544.6	1.14	271	TL	1.39	7523	197	4729.7	0.09
R111	1484	21	2318.6	1.2	1173	TL	1.61	5281	294	1732.5	0.41
R112	2044.6^{P}	34	TL	-	3126	TL	-	4220	197	TL	-

Table 19: Results for $n = 75$, $Q = 60$ and $\rho = 25\%$

	F-VRPTW					VRPTW	
			BPC-FC-PP				
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	$\Delta z_{IP}(\%)$
			$k = 2$				
R101	1552.5	20	0.9	0.72	23	0.7	0
R102	1471	19	11.1	0.89	43	8.2	0
R103	1388.1	19	828.5	1.58	1715	170.2	0.32
R104	1350.2	19	1164.3	1.04	1805	915	0
R105	1455.3	19	2.7	0.54	19	2.1	0
R106	1412.1	19	30.8	1.15	75	16.3	0
R107	1357.4	19	30.8	0.98	49	20.9	0
R108	1347.1	19	584.1	1.09	617	370.9	0
R109	1380.1	19	42.9	1.11	161	42.1	0
R110	1356.7	19	92.7	1.11	181	67.2	0
R111	1365.7	19	336.8	1.37	541	228.1	0
R112	1339.6	19	878.6	1	1209	693.6	0
			$k = 3$				
R101	1552.5	20	0.7	0.68	19	0.7	0
R102	1471	19	13.5	0.83	35	8.2	0
R103	1383.7	19	215.8	1.3	373	170.2	0
R104	1350.2	19	1341.2	1.04	1931	915	0
R105	1460.7	19	6.2	0.8	37	2.1	0.37
R106	1412.1	19	32.9	1.15	67	16.3	0
R107	1357.4	19	34	0.99	33	20.9	0
R108	1347.1	19	493.9	1.09	499	370.9	0
R109	1380.1	19	46.6	1.1	165	42.1	0
R110	1356.7	19	89.5	1.09	191	67.2	0
R111	1365.7	19	211.3	1.37	379	228.1	0
R112	1339.6	19	755.3	1	1069	693.6	0
			$k = 4$				
R101	1552.7	20	0.5	0.54	11	0.7	0.01
R102	1476.6	19	57.3	0.83	217	8.2	0.38
R103	1389.6	19	570.6	1.33	997	170.2	0.43
R104	1355.6	19	6742.9	1.19	10135	915	0.4
R105	1460.3	19	7.4	0.78	41	2.1	0.34
R106	1418.2	19	99.8	1.35	257	16.3	0.43
R107	1357.4	19	22.1	0.81	11	20.9	0
R108	1347.1	19	318.3	1.03	327	370.9	0
R109	1380.1	19	54.1	1.1	177	42.1	0
R110	1356.7	19	83.6	1.09	137	67.2	0
R111	1368.4	19	410.3	1.38	717	228.1	0.2
R112	1339.6	19	721	1	1009	693.6	0
			$k = 6$				
R101	1552.5	20	0.8	0.63	23	0.7	0
R102	1471	19	6.6	0.75	7	8.2	0
R103	1388.1	19	678.1	1.54	1109	170.2	0.32
R104	1350.2	19	1580.2	1.04	1943	915	0
R105	1460.7	19	7.3	0.76	37	2.1	0.37
R106	1412.1	19	33.4	0.97	65	16.3	0
R107	1357.4	19	45.4	0.98	49	20.9	0
R108	1347.1	19	612.9	1.09	535	370.9	0
R109	1380.1	19	47.3	1.07	151	42.1	0
R110	1356.7	19	52.4	0.99	78	67.2	0
R111	1365.7	19	232.6	1.37	369	228.1	0
R112	1339.6	19	746.2	1	1063	693.6	0

Table 20: Results for $n = 75$, $Q = 60$ and $\rho = 75\%$

	F-VRPTW					VRPTW	
			BPC-FC-PP				
Instance	z_{IP}	#Veh	T (s)	Gap (%)	#Nodes	T (s)	$\Delta z_{IP}(\%)$
			$k = 2$				
R101	1552.5	20	0.7	0.72	19	0.7	0
R102	1471	19	9.6	0.88	33	8.2	0
R103	1388.1	19	925.3	1.59	1771	170.2	0.32
R104	1350.2	19	1158.9	1.01	1783	915	0
R105	1460.2	19	3.3	0.76	23	2.1	0.34
R106	1414.1	19	55.9	0.96	123	16.3	0.14
R107	1357.4	19	26.6	0.96	35	20.9	0
R108	1347.1	19	653.1	1.09	703	370.9	0
R109	1380.1	19	57.6	1.08	237	42.1	0
R110	1356.7	19	113.4	1.11	233	67.2	0
R111	1365.7	19	235.6	1.36	425	228.1	0
R112	1339.6	19	1324	1	2035	693.6	0
			$k = 3$				
R101	1552.5	20	0.7	0.72	19	0.7	0
R102	1471	19	10.9	0.89	43	8.2	0
R103	1383.7	19	245.9	1.33	451	170.2	0
R104	1350.2	19	1173.7	1.01	1799	915	0
R105	1460.2	19	4.7	0.76	31	2.1	0.34
R106	1414.1	19	51.8	0.94	133	16.3	0.14
R107	1357.4	19	39	0.99	41	20.9	0
R108	1347.1	19	565.3	1.09	673	370.9	0
R109	1380.1	19	53.5	1.08	197	42.1	0
R110	1356.7	19	76.5	1.11	157	67.2	0
R111	1365.7	19	246	1.36	509	228.1	0
R112	1339.6	19	1069.5	1	2077	693.6	0
			$k = 4$				
R101	1552.5	20	0.5	0.69	9	0.7	0
R102	1471	19	9.8	0.77	29	8.2	0
R103	1388.1	19	835.4	1.59	1685	170.2	0.32
R104	1352.9	19	1799.8	1.03	2693	915	0.2
R105	1460.2	19	3.7	0.76	25	2.1	0.34
R106	1414.1	19	30.4	0.96	67	16.3	0.14
R107	1357.4	19	35.5	0.96	45	20.9	0
R108	1348.7	19	629.7	1	777	370.9	0.12
R109	1380.1	19	50	1.07	187	42.1	0
R110	1356.7	19	48.5	1.09	73	67.2	0
R111	1365.7	19	260.3	1.36	437	228.1	0
R112	1339.6	19	637.3	1	873	693.6	0
			$k = 6$				
R101	1555.5	20	1.4	0.75	35	0.7	0.19
R102	1471	19	10.4	0.88	33	8.2	0
R103	1388.1	19	765.4	1.59	1555	170.2	0.32
R104	1351	19	1751.1	1.07	2861	915	0.06
R105	1460.2	19	3.1	0.76	23	2.1	0.34
R106	1414.1	19	38.9	0.94	106	16.3	0.14
R107	1357.4	19	26.2	0.96	33	20.9	0
R108	1348.6	19	895.8	1.2	1005	370.9	0.11
R109	1380.1	19	41.9	1.07	165	42.1	0
R110	1356.7	19	41.8	1.09	75	67.2	0
R111	1365.7	19	192.5	1.36	369	228.1	0
R112	1339.6	19	660.5	1	1045	693.6	0

References

C. Altman. Optimisation de tournées de véhicules avec contraintes de fragilité. Master's thesis, Polytechnique Montréal, Montréal, Canada, 2017.

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

N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. Operations Research Letters, 34(1):58–68, 2006.

F. Carrabs, R. Cerulli, and M. G. Speranza. A branch-and-bound algorithm for the double travelling salesman problem with two stacks. Networks, 61(1):58–75, 2013.

T. Chabot, R. Lahyani, L. C. Coelho, and J. Renaud. Order picking problems under weight, fragility and category constraints. International Journal of Production Research, 55(21):6361–6379, 2017.

M. Cherkesly, G. Desaulniers, S. Irnich, and G. Laporte. Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks. European Journal of Operational Research, 250: 782–793, 2016.

C. Contardo, G. Desaulniers, and F. Lessard. Reaching the elementary lower bound in the vehicle routing problem with time windows. Networks, 65(1):88–99, 2015.

J.-F. Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. Operations Research, 54(3):573–586, 2006.

L. Costa, C. Contardo, and G. Desaulniers. Exact branch-price-and-cut algorithms for vehicle routing. Transportation Science, 53(4):946–985, 2019.

J.-F. Côté, C. Archetti, M. G. Speranza, M. Gendreau, and J.-Y. Potvin. A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with multiple stacks. Networks, 60(4):212–226, 2012.

G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. Transportation Science, 42(3):387–404, 2008.

G. Desaulniers, J. Desrosiers, and S. Spoorendonk. Cutting planes for branch-and-price algorithms. Networks, 58(4):301–310, 2011.

G. Desaulniers, O. Madsen, and S. Ropke. The vehicle routing problem with time windows. In P. Toth and D. Vigo, editors, Vehicle Routing: Problems, Methods, and Applications, MOS - SIAM Series on Optimization, pages 119–160. SIAM, 2 edition, 2014.

G. Fuellerer, K. F. Doerner, R. F. Hartl, and M. Iori. Metaheuristics for vehicle routing problems with three-dimensional loading constraints. European Journal of Operational Research, 201(3):751–759, 2010.

M. Gendreau, M. Iori, G. Laporte, and S. Martello. A tabu search algorithm for a routing and container loading problem. Transportation Science, 40(3):342–350, 2006.

S. Irnich and G. Desaulniers. Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, Column generation, pages 33–65. Springer, 2005.

M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. Operations Research, 56(2):497–511, 2008.

L. Junqueira, J. Oliveira, M. A. Carravilla, and R. Morabito. An optimization model for the vehicle routing problem with practical three-dimensional loading constraints. International Transactions in Operational Research, 20(5):645–666, 2013.

N. Kohl, J. Desrosiers, O. Madsen, S. M.M., and F. Soumis. 2-Path cuts for the vehicle routing problem with time windows. Transportation Science, 33(1):101–116, 1999.

M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. Operations Research, 53(6):1007–1023, 2005.

D. Pecin, C. Contardo, G. Desaulniers, and E. Uchoa. New enhancements for the exact solution of the vehicle routing problem with time windows. INFORMS Journal on Computing, 29(3):489–502, 2017a.

D. Pecin, A. Pessoa, M. Poggi, and E. Uchoa. Improved branch-cut-and-price for capacitated vehicle routing. Mathematical Programming Computation, 9(1):61–100, 2017b.

H. Pollaris, K. Braekers, A. Caris, G. K. Janssens, and S. Limbourg. Vehicle routing problems with loading constraints: state-of-the-art and future directions. OR Spectrum, 37(2):297–330, 2015.

G. Righini and M. Salani. New dynamic programming algorithms for the resource constrained elementary shortest path problem. Networks, 51(3):155–170, 2008.

R. Sadykov, E. Uchoa, and A. Pessoa. A bucket graph based labeling algorithm with application to vehicle routing. Transportation Science, (forthcoming), 2020.

Y. Tao and F. Wang. An effective tabu search approach with improved loading algorithms for the 3L-CVRP. Computers & Operations Research, 55:127–140, 2015.

C. D. Tarantilis, E. E. Zachariadis, and C. T. Kiranoudis. A hybrid metaheuristic algorithm for the integrated vehicle routing and three-dimensional container-loading problem. IEEE Transactions on Intelligent Transportation Systems, 10(2):255–271, 2009.

M. Veenstra, M. Cherkesly, G. Desaulniers, and G. Laporte. The pickup and delivery problem with time windows and handling operations. Computers & Operations Research, 77:127–140, 2017.