Les Cahiers du GERAD

Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks

M. Cherkesly, G. Desaulniers,
S. Irnich, G. Laporte

Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks

Marilène Cherkesly[a,b]

Guy Desaulniers[a,b]

Stefan Irnich[c]

Gilbert Laporte[a,d,e]

[a] *GERAD, HEC Montréal, Montréal (Québec) Canada, H3T 2A7*

[b] *Polytechnique Montréal, Montréal (Québec) Canada, H3C 3A7*

[c] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University, D-55099 Mainz, Germany*

[d] *CIRRELT, Montréal (Québec) Canada, H3C 3J7*
[e] *HEC Montréal, Montréal (Québec) Canada, H3T 2A7*

marilene.cherkesly@gerad.ca
guy.desaulniers@gerad.ca
irnich@uni-mainz.de
gilbert.laporte@hec.ca

March 2015

Abstract: This paper proposes models and algorithms for the pickup and delivery vehicle routing problem with time windows and multiple stacks. Each stack is rear-loaded and is operated in a last-in-first-out (LIFO) fashion, meaning that when an item is picked up, it is positioned at the rear of a stack. An item can only be delivered if it is in that position. This problem arises in the transportation of heavy or dangerous material where unnecessary handling should be avoided, such as in the transportation of cars between car dealers and the transportation of livestock from farms to slaughterhouses. To solve this problem, we propose two different branch-price-and-cut algorithms. The first solves the shortest path pricing problem with the multi-stack policy, while the second incorporates this policy partly in the shortest path pricing problem and generates additional inequalities to the master problem when infeasible multi-stack routes are encountered. Computational results obtained on instances derived from benchmark instances for the pickup and delivery traveling salesman problem with multiple stacks are reported, and reveal the advantage of incorporating the multi-stack policy in the pricing problem. Instances with up to 75 requests and with one, two and three stacks can be solved optimally within two hours of computational time.

Key Words: Vehicle routing with pickups and deliveries, loading constraints, column generation, branch-price-and-cut, valid inequalities.

1   Introduction

This paper proposes two branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks (PDPTWMS) and analyzes their performance. In the pickup and delivery problem, vehicles based at a depot are used to satisfy a set of requests which consists of transporting goods (or items) from a specific pickup location, where the item is loaded, to a specific delivery location, where the item is unloaded. We consider an unlimited fleet of identical vehicles with multiple homogeneous compartments of limited capacity. Each compartment is rear-loaded and is operated as a last-in-first-out (LIFO) stack, meaning that when an item is picked up, it is positioned on top of a stack. An item can only be delivered if it is on top of its stack and shifting items between stacks is not allowed. To illustrate, let 0 denote the depot, and let i^+ and i^- be the pickup and the delivery nodes associated with request i. Figure 1 depicts a route and the load of a two-stack vehicle with respect to the multi-stack policy. Each pickup and delivery location has a specified time window during which the service must start. A vehicle route is feasible if (i) the service at each location starts within the given time windows, (ii) the load in each compartment of the vehicle does not exceed its capacity, (iii) each completed requested is first picked up at its pickup location and then delivered at its corresponding delivery location, and (iv) the loading and unloading of the items respect the LIFO policy for each stack. Two types of costs are considered: a fixed cost for each vehicle used in the solution and a distance-related variable cost. The PDPTWMS consists of determining a set of least-cost feasible routes in which the number of vehicles is first minimized.

The PDPTWMS arises in the transportation of heavy or dangerous material for which unnecessary handling should be avoided. In particular, this problem is encountered in the transportation of cars between car dealers with multi-level vehicles, where each level is operated in a LIFO fashion. This problem also arises in the transportation of livestock from farms to slaughterhouses with multi-compartment vehicles, where each compartment is operated in a LIFO fashion. To the best of our knowledge, this problem has not been previously studied, but several of its variants have been investigated, namely the pickup and delivery problem (see Berbeglia et al. [7], Parragh et al. [23, 24] for surveys), the pickup and delivery problem with time windows (PDPTW) (see Ropke et al. [29], Battara et al. [6] for exact algorithms), the traveling salesman problem with pickup and delivery and LIFO loading (TSPPDL) (see Carrabs et al. [8], Cordeau et al. [12] for exact algorithms), the pickup and delivery problem with time windows and LIFO loading (PDPTWL) (see Cherkesly et al. [10] for three exact branch-price-and-cut algorithms), the double traveling salesman problem with multiple stacks (see Alba Martínez et al. [1], Lusby et al. [21], Petersen et al. [25] for exact algorithms and Petersen and Madsen [26] for a heuristic), the double vehicle routing problem with multiple stacks (see Iori and Riera-Ledesma [18] for exact algorithms), and the pickup and delivery traveling salesman problem with multiple stacks (PDTSPMS) (see Côté et al. [13, 14] for a branch-and-cut algorithm and a heuristic).

Among the algorithms proposed for the variants of the PDPTWMS, two exact algorithms stand out and constitute the basis of this research. Côté et al. [13] have developed a branch-and-cut algorithm for the PDTSPMS, for which several families of valid inequalities were proposed and tested, and instances with up to 21 requests were solved to optimality within one hour of computation time. Cherkesly et al. [10] have developed a branch-price-and-cut for the PDPTWL and have introduced three relaxations of the pricing problem, for which they developed shortest path labeling algorithms to enforce the LIFO policy. One
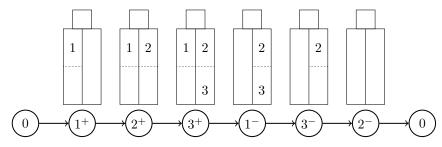

Figure 1: Route satisfying the capacity constraints and the multi-stack policy for a vehicle containing two stacks, each of capacity 2. All three items have a unit demand.

algorithm solves the pricing problem without the LIFO policy and imposes it through additional constraints in the master problem. Another algorithm solves the pricing problem with a relaxed LIFO policy and additional constraints may be added to the master problem if needed. The last one solves the pricing problem with a complete version of the LIFO policy. Computational results revealed that for harder instances the hybrid algorithm seemed to perform best.

The main objective of this paper is to develop, for the first time, different exact branch-price-and-cut algorithms for the PDPTWMS. A branch-price-and-cut algorithm is a branch-and-price algorithm in which the linear relaxation is strengthened through the generation of valid inequalities. Column generation can be adapted to constrained vehicle routing problems and have been shown to provide some of the best known results for the PDPTW (Ropke and Cordeau [28], Baldacci et al. [2]) and for the PDPTWL (Cherkesly et al. [10]).

The first algorithm developed in this paper solves the pricing problem with the multi-stack policy, whereas the second incorporates it partially in the pricing problem and generates additional inequalities to the master problem in which infeasible multi-stack routes are used in a linear relaxation solution. The results of extensive computational experiments on instances derived from known PDTSPMS instances are reported. Instances with up to 75 requests and with up to three stacks can be solved within two hours of computation time. The results show that the first algorithm always performs better than the second.

The remainder of this paper is structured as follows. Section 2 proposes a column generation formulation for the PDPTWMS and formally introduces the pricing problem. Section 3 presents our first branch-price-and-cut algorithm in which the pricing problem is solved under the multi-stack policy. Section 4 presents our second branch-price-and-cut algorithm in which the pricing problem is partly solved under the multi-stack policy and through the introduction of additional constraints in the master problem. Extensive computational results are reported in Section 5 and are followed by conclusions in Section 6.

2 A mathematical formulation

We now introduce a set partitioning formulation for the PDPTWMS. Beforehand, we provide the required notation.

2.1 Notation

Let n denote the number of requests. The PDPTWMS can be defined on a directed graph $G = (N, A)$, where $N = \{0, 1, ..., 2n, 2n+1\}$ is the set of nodes and A is the set of arcs. Nodes 0 and $2n+1$ represent two copies of the depot appearing at the start and at the end of a route, respectively. The subsets $P = \{1, ..., n\}$ and $D = \{n+1, ..., 2n\}$ are the sets of pickup and delivery nodes, respectively. With each request i is associated a pickup node $i \in P$, denoted by i^+, and a delivery node $n+i \in D$, denoted by i^-. Note that $i \in P$ refers to a pickup node and to its associated request.

With each node $i \in N$ is associated a demand q_i to be picked up or delivered, and for each request $i \in P$, $q_i > 0$ and $q_{n+i} = -q_i$. For each request $i \in P$, we refer to the load picked up at node i and delivered at node $n+i$ as an item. Moreover, we assume that $q_0 = q_{2n+1} = 0$. A time window $[\underline{w}_i, \bar{w}_i]$ is associated with each node $i \in N$, where $\underline{w}_i$ and $\bar{w}_i$ represent the earliest and the latest time at which service at node i can start, respectively. An unlimited set of identical vehicles, each with S identical stacks of capacity Q, is available. A non-negative travel cost c_{ij} and a non-negative travel time t_{ij} including the service time at node i are associated with each arc $(i, j) \in A$. The cost of each arc leaving the origin node, i.e., an arc $(0, i) \in A$, $i \in P$, also includes a large vehicle fixed cost, leading to first minimizing the number of vehicles, and then the total traveled distance. The triangle inequality is assumed to be respected for travel costs and travel times.

2.2 Set partitioning formulation

Let Ω denote the set of all feasible routes with respect to the time window constraints, the capacity constraints, and the multi-stack policy. Let c_r denote the cost of route $r \in \Omega$, i.e., a fixed vehicle cost and its total traveled

distance, and let a_{ir} be the number of times node $i \in P$ is visited in route r. Defining y_r as a binary variable equal to 1 if and only if route r is used in the solution, the PDPTWMS can be formulated as

$$\text{minimize} \quad \sum_{r \in \Omega} c_r y_r \tag{1}$$

$$\text{subject to} \quad \sum_{r \in \Omega} a_{ir} y_r = 1, \quad\quad \forall i \in P, \tag{2}$$

$$y_r \in \{0, 1\}, \quad\quad \forall r \in \Omega. \tag{3}$$

The objective function (1) minimizes the total cost and constraints (2) ensure that each request is completed exactly once. Because the model defined by (1)–(3) generally contains a large number of variables, column generation is often used to solve its linear relaxation, also called the master problem (see Desaulniers et al. [15]). At each column generation iteration, a restricted master problem (RMP) containing a subset of variables is solved through linear programming, yielding primal and dual solutions. The pricing problem is then solved to identify variables, also called columns, with negative reduced cost. In the PDPTWMS context, the pricing problem is an elementary shortest path problem with time window constraints, capacity constraints, and multi-stack policy. When such columns are identified, they are added to the RMP and a new iteration starts. Otherwise, the process stops with an optimal solution to the master problem.

2.3 Pricing problem

The column generation pricing problem aims at finding feasible routes with a negative reduced cost. In this section, we provide a formulation for this problem.

The reduced cost of arc $(i, j) \in A$ can be defined as

$$\bar{c}_{ij} = \begin{cases} c_{ij} - \alpha_i, & \forall i \in P, \\ c_{ij}, & \forall i \in N \backslash P, \end{cases} \tag{4}$$

where α_i, $i \in P$, are the dual variables associated with constraints (2).

For each node $i \in N$, let T_i be a variable representing the time at which the service begins at node i. For each arc $(i, j) \in A$, let x_{ij} be a binary variable equal to 1 if and only if arc $(i, j) \in A$ is used in the current route. For each request $i \in P$ and each stack $s \in S$, let z_i^s be a binary variable equal to 1 if and only if item i is loaded on stack s in the current route, and let Q_i^s be a variable representing the total load of stack s after leaving node i. In order to define the multi-stack policy, we introduce the following notation. Let $R = (i_1, \ldots, i_\rho)$ be a path such that $i_1 \neq 0$ and $i_\rho \neq 2n+1$ and let $N(R) = \{i_1, \ldots, i_\rho\}$ be its corresponding set of nodes. Let $\left(s_i^e\right)_{i \in N(R)}$ be an assignment (indexed by e) to the stacks in S of the items picked up or delivered in R, i.e., $s_i^e \in S$ indicates the stack to which item i is assigned. Some of these assignments might yield an infeasible path with respect to the multi-stack policy. Denote by $I(R)$ the set of infeasible item-to-stack assignments for path R. Let $\mathcal{R}$ be the set of all paths R that can have infeasible assignments, i.e., such that $I(R) \neq \emptyset$. The pricing problem for the PDPTWMS can then be modeled as

$$\text{minimize} \quad \sum_{(i,j) \in A} \bar{c}_{ij} x_{ij} \tag{5}$$

$$\text{subject to} \quad \sum_{j \in N | (i,j) \in A} x_{ij} - \sum_{j \in N | (n+i,j) \in A} x_{n+i,j} = 0, \quad \forall i \in P, \tag{6}$$

$$\sum_{j \in P} x_{0j} = 1, \tag{7}$$

$$\sum_{j \in N | (j,i) \in A} x_{ji} - \sum_{j \in N | (i,j) \in A} x_{ij} = 0, \quad \forall i \in P \cup D, \tag{8}$$

$$\sum_{i \in D} x_{i,2n+1} = 1, \tag{9}$$

$$\sum_{\mu=1}^{\rho-1} x_{i_\mu,i_{\mu+1}} + \sum_{i\in P\cap N(R)} z_i^{s_i^e} + \sum_{i\in D\cap N(R)} z_{i-n}^{s_{i-n}^e} \le 2|N(R)|-2, \quad \forall R=(i_1,...,i_\rho)\in\mathcal{R}, e\in I(R) \qquad (10)$$

$$\sum_{j\in N|(i,j)\in A} x_{ij} = \sum_{s\in S} z_i^s, \quad \forall i\in P, \qquad (11)$$

$$Q_j^s \ge Q_i^s x_{ij} + q_j x_{ij} z_j^s, \quad \forall s\in S, (i,j)\in A \text{ such that } j\in P, \quad (12)$$

$$Q_j^s \ge Q_i^s x_{ij} + q_j x_{ij} z_{j-n}^s, \quad \forall s\in S, (i,j)\in A \text{ such that } j\in D, \quad (13)$$

$$\max\{0,q_i\}z_i^s \le Q_i^s \le \min\{Q,Q+q_i\}z_i^s, \quad \forall i\in P, s\in S, \qquad (14)$$

$$T_j \ge (T_i + t_{ij})x_{ij}, \quad \forall(i,j)\in A, \qquad (15)$$

$$\underline{w}_i \le T_i \le \bar{w}_i, \quad \forall i\in N, \qquad (16)$$

$$T_i + t_{i,n+i} \le T_{n+i}, \quad \forall i\in P, \qquad (17)$$

$$x_{ij} \in \{0,1\}, \quad \forall(i,j)\in A, \qquad (18)$$

$$z_i^s \in \{0,1\}, \quad \forall i\in P, s\in S. \qquad (19)$$

The objective function (5) minimizes the sum of reduced costs. Constraints (6) ensure that the pairing constraints are respected, i.e., the pickup and delivery nodes of a request are visited in the same route. Constraints (7)–(9) define a path structure for each route. More specifically, constraints (7) and (9) ensure that each route starts and ends at the depot, while (8) are flow conservation constraints. The multi-stack policy is imposed through constraints (10) which are stated through infeasible path inequalities. We note that Côté et al. [13] have proposed an alternate way of formulating these constraints. Constraints (11) state that each picked up item must be assigned to exactly one of the stacks. Constraints (12) and (13) compute the load variables according to the arcs used in the solution and constraints (14) ensure that the capacity of each stack is respected. Constraints (15) and (16) compute the time variables and ensure that the time windows are respected. Constraints (17) impose the precedence constraints, i.e., for each request i the pickup node must be visited before the delivery node. The model is non-linear because of constraints (12)–(15), but can be linearized (see Ropke et al. [29], Côté et al. [13]).

Note that constraints (10) can be replaced by the smaller set of constraints

$$\sum_{\mu=1}^{\rho-1} x_{i_\mu,i_{\mu+1}} \le |N(R)|-2, \qquad \forall R=(i_1,...,i_\rho)\in\mathcal{R}^*, \qquad (20)$$

where $\mathcal{R}^* \subseteq \mathcal{R}$ is a subset of infeasible paths such that, for each path, there exists no feasible assignment of the items to the stacks. Solving model (5)–(9), (11)–(19), (20) could yield an infeasible solution given the values taken by the z_i^s variables at optimality, i.e., these values would provide an infeasible item-to-stack assignment for at least one path in $\mathcal{R} \setminus \mathcal{R}^*$, but there exists an alternative feasible solution having the same cost with different z_i^s values.

3 A branch-price-and-cut algorithm with multi-stack feasible paths

Our first branch-price-and-cut algorithm fully enforces the multi-stack policy in the pricing problem. In this section, we first present path relaxations and labeling algorithms for the corresponding pricing problem. We then discuss valid inequalities for the PDPTWMS and branching strategies.

3.1 Path relaxations and labeling algorithms

The pricing problem is an elementary shortest path problem with pickups and deliveries, time windows, capacity constraints, and multi-stack policy. It can be solved through a labeling algorithm. A label stores information about a partial path starting at the origin node and ending at some node η. Each element stored in a label is called a component. Starting from an initial label E_0 at the origin node 0, a labeling algorithm propagates labels toward the destination node with resource extension functions. To avoid enumerating all feasible paths, some labels are eliminated through a dominance criterion.

The pricing problem can be relaxed by allowing cycles in paths, that is, a request can be completed more than once. These relaxations usually yield weaker master problem lower bounds. Paths with cycles cannot be part of a feasible integer solution, hence branching ensures that the final solution contains only elementary paths.

Sections 3.1.1 and 3.1.2 describe labeling algorithms for the elementary and non-elementary versions of the pricing problem, respectively.

3.1.1 Elementary shortest path problem with pickups and deliveries, time windows, capacity constraints and multi-stack policy

The first version of the constrained shortest path problem respects the elementarity constraints. The ideas presented in this section are non-trivial extensions of those initially proposed by Cherkesly et al. [10] for the PDPTWL. For a given label E, the following components are stored:

- $\eta(E)$, the end node of the partial path;
- $t(E)$, the start of service time at node η;
- $c(E)$, the cumulated reduced cost;
- $U(E)$, the set of unreachable requests;
- $l_i(E), \forall i \in P$, the load in the stack under item $i \in P$ (including its own load);
- $\mathcal{S}_{ij}(E), \forall i, j \in P$, the relative positions of items i and j in a given stack;
- $\mathcal{C}_{ij}(E), \forall i, j \in P$, the concurrent presence of items i and j in different stacks.

A request $i \in P$ is said to be unreachable if i has already been visited on the partial path, or if traveling directly from η to i violates the time window constraints at node $i \in P$. For a given label E, let $R(E) = (0, i_1, i_2, ..., i_\rho = \eta(E))$ be the partial path represented by this label. Then

$$U(E) = \{i \in P | i \in R(E)\} \cup \{i \in P | t(E) + t_{\eta(E),i} > \bar{w}_i\}. \tag{21}$$

The relative position of two items $i, j \in P$ in a given stack indicates that the two items are simultaneously in the same stack and item i is on top of item j, that is

$$\mathcal{S}_{ij}(E) = \begin{cases} 1 & \text{if } i = j \text{ and item } i \text{ is in the vehicle,} \\ 1 & \text{if item } i \text{ is in the same stack as item } j \text{ and on top of it} \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

Moreover, for any two items $i, j \in P$, we need to know whether both are simultaneously onboard and in different stacks, or not, that is

$$\mathcal{C}_{ij}(E) = \begin{cases} 1 & \text{if items } i \text{ and } j \text{ are simultaneously in the vehicle} \\ & \qquad \text{but not in the same stack,} \\ 0 & \text{otherwise.} \end{cases} \tag{23}$$

This new notation is as powerful as the notation proposed by Cherkesly et al. [10] for the single-stack case, but is better suited for the multi-stack variant because it eliminates the symmetry between the S identical stacks.

Given a label E, its extension along an arc $(\eta(E), j) \in A$ is allowed if one of the following three conditions holds:

$$0 < j \leq n \qquad \text{and} \qquad j \notin U(E), \tag{24}$$
$$n < j \leq 2n \qquad \text{and} \qquad \mathcal{S}_{j-n,j-n}(E) = 1 \qquad \text{and} \qquad \mathcal{S}_{i,j-n}(E) = 0, \forall i \in P \backslash \{j-n\}, \tag{25}$$
$$j = 2n+1 \qquad \text{and} \qquad \mathcal{S}_{ii}(E) = 0, \forall i \in P. \tag{26}$$

Condition (24) ensures that if j is a pickup node, then it must not have been previously visited, and must be reachable with respect to time windows. Condition (25) ensures that if j is a delivery node, then its

corresponding item must be on top of one of the stacks, i.e., the item is in the vehicle and there is no item on top. Condition (26) ensures that if j is the destination node, then all the picked items on the path must have been delivered. Together these three conditions ensure that each request will be completed at most once for any complete path from 0 to $2n+1$.

If the extension to a pickup node is allowed, i.e., condition (24) is fulfilled, then several new labels may result. Indeed, one new label per stack in use can be created. Thus, for a given label E, we define

$$\mathcal{H}(E) = \{i \in P | \mathcal{S}_{ii}(E) = 1 \text{ and } \mathcal{S}_{ji}(E) = 0, \forall j \in P \backslash \{i\}\} \tag{27}$$

as the top items in the stacks. In particular, $|\mathcal{H}(E)|$ is the number of stacks currently in use. If this number is less than the number of stacks, i.e., $|\mathcal{H}(E)| < S$, then there exists at least one empty stack. In such a case, in order to allow the addition of an item on top of an empty stack, an additional auxiliary top item $h = 0$ is created. We define

$$\mathcal{H}_0(E) = \begin{cases} \mathcal{H}(E) \cup \{0\} & \text{if } |\mathcal{H}(E)| < S, \\ \mathcal{H}(E) & \text{otherwise.} \end{cases} \tag{28}$$

In summary, for the extension to a pickup node $j \in P$, one new label E^h for each $h \in \mathcal{H}_0(E)$ is created. If j is a delivery node, $j \in D$, a single new label E^h for $h = j - n$ is generated. If j is a pickup node and all stacks are empty, i.e., $\mathcal{H}(E) = \emptyset$, a single new label E^h is created for the auxiliary top item $h = 0$. Thus, given a new label E, an arc $(\eta(E), j)$, and a top item $h \in \mathcal{H}_0(E)$ for $j \in P$, or $h = j - n$ for $j \in D$, the extension is computed as follows:

$$\eta(E^h) = j, \tag{29}$$

$$t(E^h) = \max\{t(E) + t_{\eta(E),j}, \underline{w}_j\}, \tag{30}$$

$$c(E^h) = c(E) + \bar{c}_{\eta(E),j}, \tag{31}$$

$$U(E^h) = \begin{cases} U(E) \cup \{j\} \cup \{i \in P | t(E^h) + t_{\eta(E^h),i} > \bar{w}_i\} & \text{if } j \in P, \\ U(E) \cup \{i \in P | t(E^h) + t_{\eta(E^h),i} > \bar{w}_i\} & \text{if } j \in D, \end{cases} \tag{32}$$

$$l_m(E^h) = \begin{cases} l_h(E) + q_m & \text{if } j \in P, \ j = m, \\ 0 & \text{if } j \in D, \ m = j - n, \\ l_m(E) & \text{otherwise,} \end{cases} \qquad \forall m \in P, \tag{33}$$

$$\mathcal{S}_{mi}(E^h) = \begin{cases} 1 & \text{if } j \in P, \ j = m, \ \mathcal{S}_{hi}(E) = 1, \\ 1 & \text{if } j \in P, \ m = i = j, \\ 0 & \text{if } j \in D, \ j - n = m, \\ \mathcal{S}_{mi}(E) & \text{otherwise,} \end{cases} \qquad \forall i, m \in P, \tag{34}$$

$$\mathcal{C}_{mi}(E^h) = \begin{cases} 1 & \text{if } j \in P, \ j = m, \ \mathcal{C}_{hi}(E) = 1, \\ 1 & \text{if } j \in P, \ i = j, \ \mathcal{C}_{hm}(E) = 1, \\ 0 & \text{if } j \in D, \ j - n = m \text{ or } j - n = i, \\ \mathcal{C}_{mi}(E) & \text{otherwise,} \end{cases} \qquad \forall i, m \in P. \tag{35}$$

Equations (33) state that if j is a pickup node, then the load under it must be the total load in the chosen stack, plus its own load; if j is a delivery node, then the load under item $j - n$ is 0 because item $j - n$ is no longer in the vehicle, and otherwise the load under each item remains the same. Equations (34) update the positions of items that are in the same stack. If j is a pickup node, then it must be on top of all nodes below h, on top of node h, and must be in the vehicle. If j is a delivery node, then there are no more items below $j - n$. The other positions are unchanged. Equations (35) update the information about items that are simultaneously in the vehicle but are not in the same stack. If j is a pickup node, then it must be separated from all nodes that are not in the same stack as node h, and if j is a delivery node then there are no more items simultaneously onboard with $j - n$ and in different stacks. The other positions remain the same.

A new label E^h is kept if it respects the time windows and the capacity constraints, that is, if

$$t(E^h) \leq \bar{w}_j, \tag{36}$$
$$l_h(E^h) \leq Q. \tag{37}$$

Finally, a label E_1 dominates a label E_2 if

$$\eta(E_1) = \eta(E_2), \tag{38}$$
$$t(E_1) \leq t(E_2), \tag{39}$$
$$c(E_1) \leq c(E_2), \tag{40}$$
$$U(E_1) \subseteq U(E_2), \tag{41}$$
$$\mathcal{S}_{ij}(E_1) \leq \mathcal{S}_{ij}(E_2), \qquad\qquad \forall i, j \in P, \tag{42}$$
$$\mathcal{C}_{ij}(E_1) \leq \mathcal{C}_{ij}(E_2), \qquad\qquad \forall i, j \in P. \tag{43}$$

Conditions (38)–(42) constitute a valid dominance criterion for the single-stack case, i.e., the PDPTWL (see Cherkesly et al. [10]), if the delivery triangle inequality holds. Note that the definition of $\bar{c}_{ij}$ in formula (4) does not necessarily ensure the delivery triangle inequality $\bar{c}_{ij} + \bar{c}_{jk} \geq \bar{c}_{ik}$ for all delivery nodes $j \in D$. In this situation, Ropke and Cordeau [28] propose a procedure to transform an arbitrary cost matrix into a cost matrix that satisfies the delivery triangle inequality. We apply the same procedure before solving the pricing problem.

However, the single-stack dominance criterion is not valid for the multi-stack variant. In fact, without conditions (43), the dominance criterion would not be valid because items in the same stack for label E_2 could be in different stacks for label E_1. In such a case, the possible extensions of label E_2 could be infeasible for label E_1 with respect to capacity constraints, yielding wrongly dominated labels. Figure 2 depicts such a case where $Q = 2$, $q_1 = 1$, $q_2 = 1$, and $q_3 = 2$. Figure 2a and 2b illustrate the configuration of the vehicle for labels E_1 and E_2, respectively. One can see that item 3 cannot be loaded with the first configuration, but can be with the second one. In that case, conditions (38)–(42) are respected. Conditions (43) are then necessary to allow a proper dominance criterion.

Proposition 3.1 *Conditions* (38)–(43) *constitute a valid dominance criterion whenever* $\bar{c}_{ij}$ *satisfies the delivery triangle inequality.*

Proof. The proof is similar to that of Proposition 3 in Cherkesly et al. [10]. We show that for every feasible completion of E_2 there exists a feasible completion of E_1 with no greater reduced cost. Let r be a path extending $R(E_2)$ to node $2n + 1$ such that $(R(E_2), r)$ is feasible with respect to time windows, elementarity constraints, pickup and delivery constraints, capacity constraints and the multi-stack policy. If no such path exists, then clearly one can remove label E_2. Let r' be the path obtained from r by removing the deliveries for each request $i \in P$ with $\mathcal{S}_{ii}(E_1) = 0$ and $\mathcal{S}_{ii}(E_2) = 1$. Because $(R(E_2), r)$ is feasible with respect to elementarity constraints, and pickup and delivery constraints, then so is $(R(E_1), r')$. Because the triangle inequality is assumed for travel times and $(R(E_2), r)$ is feasible with respect to the time windows, then so is $(R(E_1), r')$. The capacity constraints for each stack are not violated because items in different stacks for

(a) Stack configuration for label E_1: $\mathcal{S}_{11}(E_1) = \mathcal{S}_{22}(E_1) = 1$ and $\mathcal{C}_{12}(E_1) = \mathcal{C}_{21}(E_1) = 1$

(b) Stack configuration for label E_2: $\mathcal{S}_{11}(E_2) = \mathcal{S}_{22}(E_2) = \mathcal{S}_{21}(E_2) = 1$ and $\mathcal{C}_{12}(E_2) = \mathcal{C}_{21}(E_2) = 0$
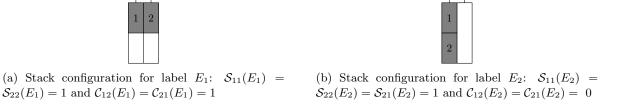
Figure 2: Example where two labels cannot be compared, $q_1 = q_2 = 1$, $q_3 = 2$ and $Q = 2$

E_1 are also in different stacks for E_2, and items in the same stack for E_1 are also in the same stack for E_2, thus the capacity constraints are respected on $(R(E_1), r')$. The multi-stack policy is not violated because the order in which the deliveries are performed on $(R(E_1), r')$ is the same as on $(R(E_2), r)$. Because $(R(E_2), r)$ is feasible, then so is $(R(E_1), r')$. Because the delivery triangle inequality holds for the reduced cost component c, the cost of r' does not exceed that of r. Thus, $c(E_1) \leq c(E_2)$ implies that the cost of $(R(E_1), r')$ is at most equal to that of $(R(E_2), r)$. Hence, label E_1 dominates label E_2. $\qquad\square$

3.1.2 Shortest path problem with pickups and deliveries, time windows, capacity constraints and multi-stack policy

The second version of the constrained shortest path problem allows paths to contain cycles under the following two conditions:

(i) a pickup cannot be performed again before its corresponding delivery has been completed;

(ii) the precedence constraints for every request must be respected.

In this version of the algorithm, a label E stores the components $\eta(E), t(E), c(E), l_i(E), i \in P, \mathcal{S}_{ij}(E)$, and $\mathcal{C}_{ij}(E)$, $i, j \in P$. The extension of a label E along arc $(\eta(E), j)$ is allowed if E and j satisfy condition (25), (26), or

$$0 < j \leq n \quad \text{and} \quad \mathcal{S}_{jj}(E) = 0. \tag{44}$$

A label E^h is then created for each top item $h \in \mathcal{H}_0(E)$ using the extension function (29)–(31) and (33)–(35). Condition (44) relaxes condition (24) by allowing cycles to occur while forbidding to pick up the same request twice without delivering it in the meantime. The resulting label E^h is kept if it satisfies the time windows (36) and the capacity constraints (37) at node j. If the delivery triangle inequality holds, then the following dominance criterion is valid: a label E_1 dominates a label E_2 if conditions (38)–(40), and (42)–(43) are respected.

The reader can easily adapt the arguments of Proposition 3.1 for this version of the shortest path problem.

3.2 Valid inequalities

We now present valid inequalities commonly used to solve the PDPTW and applicable to the PDPTWMS. These are 2-path cut inequalities, rounded capacity inequalities, and subset-row inequalities. We also present a family of cuts based on the branching on the number of vehicles. These inequalities are added within the master problem. For the sake of conciseness, we omit the discussion about the impact on the reduced cost of adding such inequalities (see Desaulniers et al. [16]). These inequalities are used for both branch-price-and-cut algorithms.

If the number of vehicles is fractional, two branches are created: $\sum_{r \in \Omega} y_r \leq \lfloor \sum_{r \in \Omega} \tilde{y}_r \rfloor$ and $\sum_{r \in \Omega} y_r \geq \lceil \sum_{r \in \Omega} \tilde{y}_r \rceil$, where $(\tilde{y}_1, \dots, \tilde{y}_{|\Omega|})$ is the computed fractional solution of the master problem. Because we first minimize the number of vehicles, the number of vehicles used in the solution of the master problem is a lower bound on the number of vehicles used in the optimal integer solution. In this case, the inequality

$$\sum_{r \in \Omega} y_r \geq \left\lceil \sum_{r \in \Omega} \tilde{y}_r \right\rceil \tag{45}$$

is added to the master problem and replaces the branching on the number of vehicles.

Kohl et al. [20] have introduced 2-path cuts in the context of the vehicle routing problem with time windows (VRPTW). These were later shown to be valid for the PDPTW (see Ropke and Cordeau [28]). Let $N_S \subseteq P \cup D$ be a subset of nodes that cannot be served by a single vehicle and let $\delta(N_S) = \{(i, j) \in A | i \in N_S, j \in N \backslash N_S\}$ represent the set of arcs exiting set N_S. Then the inequality

$$\sum_{r \in \Omega} \sum_{(i,j) \in \delta(N_S)} b_{ij}^r y_r \geq 2 \tag{46}$$

is valid, where b_{ij}^r is a constant equal to the number of times arc $(i, j) \in A$ is used in route r. Identifying a subset of nodes that cannot be served by a single vehicle means determining whether the corresponding pickup and delivery traveling salesman problem with time windows is feasible. Because this is an NP-complete problem (see Savelsbergh [30]), the separation of violated 2-path cuts is achieved by means of a greedy heuristic (see Ropke and Cordeau [28]).

The rounded capacity inequalities are often used for the vehicle routing problem (VRP), the VRPTW, and the PDPTW (see, e.g., Naddef and Rinaldi [22], Cordeau [11], Ropke et al. [29]) and have been adapted to the PDTSPMS (see Côté et al. [13]). Let $N_S \subseteq P \cup D$ be a subset of nodes and let $\xi(N_S)$ be a lower bound on the number of vehicles needed to visit all nodes in N_S. Then, the inequality

$$\sum_{r \in \Omega} \sum_{(i,j) \in \delta(N_S)} b_{ij}^r y_r \geq \xi(N_S) \tag{47}$$

is valid for $\xi(N_S) = \max \left\{ 1, \left\lceil \frac{q(\pi(N_S))}{SQ} \right\rceil, \left\lceil \frac{-q(\sigma(N_S))}{SQ} \right\rceil \right\}$, where SQ is the total capacity of the vehicle, $\pi(N_S) = \{i \in P | i \notin N_S, n+i \in N_S\}$ denotes the set of predecessors of N_Ss and $\sigma(N_S) = \{n+i \in D | i \in N_S, n+i \notin N_S\}$ denotes the set of successors of N_S. The lower bound on the load of the vehicles entering N_S is $q(\pi(N_S)) = \sum_{i \in \pi(N_S)} q_i$, and the lower bound on the load of the vehicles leaving N_S is $q(\sigma(N_S)) = \sum_{n+i \in \sigma(N_S)} q_i$. These inequalities are separated by means of a heuristic enumerative procedure (see Ropke et al. [29]).

The subset-row inequalities were introduced by Jepsen et al. [19] for the VRPTW and are a special case of the clique inequalities. These inequalities are the rank-1 Chvátal-Gomory inequalities defined as

$$\sum_{r \in \Omega} \left\lfloor \frac{1}{\chi} \sum_{i \in N_S} a_{ir} \right\rfloor y_r = \left\lfloor \frac{|N_S|}{\chi} \right\rfloor, \quad \forall N_S \subseteq P, 2 \leq \chi \leq |N_S|, \tag{48}$$

where N_S is a subset of pickup nodes. As in Jepsen et al. [19] and Desaulniers et al. [17], we focus on the inequalities defined for subsets of three customers because these can be efficiently separated. These subset-row inequalities can be rewritten as

$$\sum_{r \in \Omega_S} y_r \leq 1, \quad \forall N_S \subseteq P \text{ such that } |N_S| = 3, \tag{49}$$

where $\Omega_S \subseteq \Omega$ is the subset of routes completing at least two requests in N_S. Because handling the dual prices of the active subset-row inequalities in the subproblem can be highly time-consuming, we limit their usage by generating them only in the first two levels of the branching tree and adding at most 50 cuts at once.

3.3 Branching

In a branch-price-and-cut algorithm, branching is used to obtain integer feasible solutions and should be compatible with the column generation process, especially with the algorithm used to solve the pricing problem. With the dominance criterion (38)–(43), the removal of arcs must preserve the delivery triangle inequality (see Ropke and Cordeau [27]). Consequently, we propose to branch on the outflow of node subsets (see Naddef and Rinaldi [22]). This branching strategy adds constraints to the master problem, yielding additional dual prices to be incorporated into the objective function of the pricing problem (see Desaulniers et al. [16]). In this branching strategy, a subset of nodes N_S is selected such that $f(N_S) = \sum_{r \in \Omega} \sum_{(i,j) \in \delta(N_S)} b_{ij}^r \tilde{y}_r$ is as far as possible from the nearest integer, where $f(N_S)$ is the total outflow for the set N_S of the computed fractional solution of the master problem. Two branches are then created by adding the following constraints to the master problem associated with each branch:

$$\sum_{r \in \Omega} \sum_{(i,j) \in \delta(N_S)} b_{ij}^r y_r \leq \lfloor f(N_S) \rfloor, \tag{50}$$

$$\sum_{r \in \Omega} \sum_{(i,j) \in \delta(N_S)} b_{ij}^r y_r \geq \lceil f(N_S) \rceil. \tag{51}$$

The exploration of the enumeration tree is achieved through a best-first strategy.
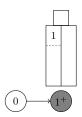
4   A branch-price-and-cut algorithm with relaxed multi-stack paths

This second branch-price-and-cut algorithm deals with the multi-stack policy partly in the set partitioning formulation and partly in the master problem. This pricing problem is easier to solve, but the extended set partitioning formulation is weaker yielding worse lower bounds. As Cherkesly et al. [10] did in their hybrid branch-price-and-cut algorithm for the PDPTWL, we solve the shortest path problem under relaxed multi-stack constraints, i.e., the LIFO policy must be respected for the last κ items of each compartment. An ejection process is therefore needed: when a pickup node is visited, its corresponding item is put on top of a stack; if the height of the stack exceeds κ, the lowest item is ejected from the stack but is kept in the corresponding compartment. Thus, a compartment can contain stacked items, for which the extension of the partial path needs to respect the LIFO policy, and ejected items, for which the extension of the partial path does not need to respect the LIFO policy. There is no given ordering for the delivery of the ejected items, but these can only be delivered if there are no more stacked items in the compartment. Corresponding infeasible path inequalities are added to the master problem when infeasible multi-stack routes are used in a linear relaxation solution. Figure 3 presents an example, for $\kappa = 1$, in which the path must respect the LIFO policy for the items in grey. The vehicle contains two compartments and each compartment has a corresponding stack with a maximal size of one item. Note that item 2 is ejected from the second stack in Figure 3c and item 3 is ejected from the second stack in Figure 3d because the maximal size is reached.
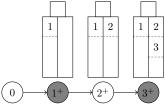
4.1   Labeling algorithm

We now describe the modifications to the labeling algorithm presented in Section 3 that we have implemented to handle this variant. The valid inequalities and the branching decisions used are those of Sections 3.2 and 3.3.
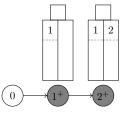
In the elementary version of the problem, a label E stores the components $\eta(E)$, $t(E)$, $c(E)$, $U(E)$, $l_i(E)$, $i \in P$, $\mathcal{C}_{ij}(E)$, and $\mathcal{S}_{ij}^{EP}(E)$, $i, j \in P$. $\mathcal{S}_{ij}^{EP}(E)$ is a relaxation of $\mathcal{S}_{ij}(E)$ that considers an ejection process.
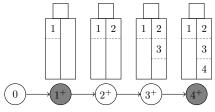
(a) Label E^a: the first stack contains item 1, thus the extension of the path must respect the LIFO policy for item 1, $\mathcal{S}_{11}^{EP}(E^a) = 1$

(b) Label E^b: the first stack contains item 1 and the second stack contains item 2, thus the extension of the path must respect the LIFO policy for items 1 and 2, $\mathcal{S}_{11}^{EP}(E^b) = \mathcal{S}_{22}^{EP}(E^b) = 1$

(c) Label E^c: the first stack contains item 1, the second stack contains item 3, and the second compartment contains items 2 and 3, thus the extension of the path does not need to respect the LIFO policy for item 2 as the maximal size of the LIFO stack has been reached for the second stack, $\mathcal{S}_{11}^{EP}(E^c) = \mathcal{S}_{22}^{EP}(E^c) = \mathcal{S}_{33}^{EP}(E^c) = \mathcal{S}_{32}^{EP}(E^c) = 1$

(d) Label E^d: the first stack contains item 1, the second stack contains item 4, and the second compartment contains items 2, 3, and 4, thus the extension of the path does not need to respect the LIFO policy for items 2 and 3, $\mathcal{S}_{11}^{EP}(E^d) = \mathcal{S}_{22}^{EP}(E^d) = \mathcal{S}_{33}^{EP}(E^d) = \mathcal{S}_{44}^{EP}(E^d) = \mathcal{S}_{42}^{EP}(E^d) = \mathcal{S}_{43}^{EP}(E^d) = 1$ but $\mathcal{S}_{23}^{EP}(E^d) = \mathcal{S}_{32}^{EP}(E^d) = 0$

Figure 3: The extension of the path must respect the LIFO policy for the items in grey ($\kappa = 1$)

For a given label E, $\mathcal{S}_{ij}^{EP}(E)$ is defined as

$$
\mathcal{S}_{ij}^{EP}(E) = \begin{cases}
1 & \text{if } i = j \text{ and item } i \text{ is in the vehicle,} \\
1 & \text{if items } i \text{ and } j \text{ are in the same stack and } i \text{ is on top of } j, \\
1 & \text{if items } i \text{ and } j \text{ are in the same compartment,} \\
& \quad i \text{ is in the stack, and } j \text{ is ejected,} \\
0 & \text{if items } i \text{ and } j \text{ are in the same compartment and both are ejected,} \\
0 & \text{otherwise.}
\end{cases}
\tag{52}
$$

That is, if two items $i \in P$ and $j \in P$, $i \neq j$, are in the same vehicle compartment, request i is in the stack and node i^+ was visited after node j^+, then $\mathcal{S}_{ij}^{EP}(E) = 1$ and $\mathcal{S}_{ji}^{EP}(E) = 0$. If two items $i \in P$ and $j \in P$, $i \neq j$, are in the same vehicle compartment but none of them are in the stack, i.e., if both have been ejected, then $\mathcal{S}_{ij}^{EP}(E) = \mathcal{S}_{ji}^{EP}(E) = 0$.

The extension of a label E along an arc $(\eta(E), j) \in A$ is allowed if it satisfies one of the three conditions, (24)–(26), where $\mathcal{S}_{ij}(E)$ is replaced with $\mathcal{S}_{ij}^{EP}(E)$, $\forall i, j \in P$.

Defining $\mathcal{H}(E)$ as in equation (27), $\mathcal{H}(E)$ contains all items that can be delivered, i.e., those that have not been ejected from a stack and are on top of a stack, and those that have been ejected from a stack and are not under an item for which the multi-stack policy needs to be respected. In order to have at most one extension per compartment, the top items are defined as the items for which one of these two conditions is respected:

(i) the item is in the stack and no item is on top of it;

(ii) the item is not in the stack, and is in a compartment that has no item in its stack. One arbitrary item is kept to represent each non-empty compartment. In the following, we choose the item with the smallest index.

We define $\mathbb{C}(E)$ as the set of top items for the relaxed multi-stack policy which can be computed as

$$
\mathbb{C}(E) = \{i \in \mathcal{H}(E) | \mathcal{C}_{ij}(E) = 1, j < i, j \in \mathcal{H}(E)\}.
\tag{53}
$$

Note that if $|\mathbb{C}(E)| < S$, an additional auxiliary top item is added in order to allow loading on top of empty stacks. Thus, we define

$$
\mathbb{C}(E)_0 = \begin{cases}
\mathbb{C}(E) \cup \{0\} & \text{if } |\mathbb{C}(E)| < S, \\
\mathbb{C}(E) & \text{otherwise.}
\end{cases}
\tag{54}
$$

The extension of a label E along an arc $(\eta(E), j)$ will create a new label E^h, $\forall h \in \mathbb{C}(E)$. Note that in some cases only one label is created (as explained in Section 3.1.1). For each label E^h, $\mathcal{O}(E^h)$ is defined as the open requests in the vehicle that have been ejected from the stack, i.e., requests that are currently onboard but for which the extension of the label does not need to respect the multi-stack policy. $\mathcal{O}(E^h)$ is computed as

$$
\mathcal{O}(E^h) = \left\{ i \in P \middle| \mathcal{C}_{hi}(E) = 0 \text{ and } \left(\sum_{j \in P} \mathcal{S}_{ji}^{EP}(E) \geq \kappa \text{ or} \right.\right.
$$
$$
\left.\left. \exists j \in P \backslash \{i\} \text{ such that } \mathcal{S}_{jj}^{EP}(E) = 1, \mathcal{C}_{ij}(E) = 0, \mathcal{S}_{ij}^{EP}(E) = \mathcal{S}_{ji}^{EP}(E) = 0 \right) \right\}.
\tag{55}
$$

Equation (55) states that $i \in P$ is an open request with respect to top item $h \in \mathbb{C}(E)_0$ if i and h are in the same vehicle compartment, and if there are at least κ items between the positions of i and h or if i has already been ejected from the stack.

The components of label E^h are set with equations (29)–(32), (35) and

$$
l_m(E^h) = \begin{cases} l_h(E) + q_j & \text{if } j \in P, \ j = m, \\ \sum_{i \in \mathcal{O}(E^h)} q_i & \text{if } j \in P, \\ 0 & \text{if } j \in D, \ m = j - n, \\ l_m(E) & \text{otherwise,} \end{cases} \qquad \forall m \in \mathcal{O}(E^h), \qquad (56)
$$

$$
\mathcal{S}_{mi}^{EP}(E^h) = \begin{cases} 1 & \text{if } j \in P, j = m, \ \forall i \in P \text{ such that } \mathcal{S}_{hi}^{EP}(E) = 1, \\ 1 & \text{if } j \in P, j = m, \ \forall i \in \mathcal{O}(E^h), \\ 1 & \text{if } j \in P, i = m = j, \\ 0 & \text{if } j \in P, \\ 0 & \text{if } j \in D, j - n = m, \ \forall i \in P, \\ \mathcal{S}_{mi}^{EP}(E) & \text{otherwise.} \end{cases} \qquad \forall m, i \in \mathcal{O}(E^h), \qquad (57)
$$

Equation (56) replaces equation (33); if two requests are in the same compartment but none of them are in the stack, their total loads will be the same. Equation (57) updates the positions of the items in the compartment. If j is a pickup node then it is on top of the stack, i.e., on top of all other items in the compartment, either in the stack or ejected. If the stack has reached its maximal size, then no order is imposed among all ejected items. If j is a delivery node, then they are no more items linked to $j - n$. Furthermore, label E^h is kept if it respects the time window constraints (36) and the capacity constraints (37).

Finally, a label E_1 dominates a label E_2 if conditions (38)–(41), (43) and

$$
\mathcal{S}_{ij}^{EP}(E_1) \le \mathcal{S}_{ij}^{EP}(E_2), \qquad \forall i, j \in P, \qquad (58)
$$

hold.
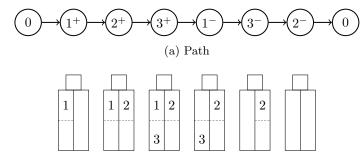
Proposition 4.1 *Conditions* (38)–(41), (43) *and* (58) *constitute a valid dominance criterion whenever* $\bar{c}_{ij}$ *satisfies the delivery triangle inequality.*

Proof. The proof is similar to that of Proposition 3.1. We show that for every feasible completion of E_2 there exists a feasible completion of E_1 with no larger reduced cost. Let r be a path extending $R(E_2)$ to node $2n + 1$ such that $(R(E_2), r)$ is feasible with respect to time windows, elementarity constraints, pickup and delivery constraints, capacity constraints and relaxed multi-stack policy. If no such path exists, then clearly one can remove label E_2. Let r' be the path obtained from r by removing the deliveries for each request $i \in P$ with $\mathcal{S}_{ii}^{EP}(E_1) = 0$ and $\mathcal{S}_{ii}^{EP}(E_2) = 1$. Because $(R(E_2), r)$ is feasible with respect to elementarity constraints, and pickup and delivery constraints, then so is $(R(E_1), r')$. Because the triangle inequality is assumed for travel times and $(R(E_2), r)$ is feasible with respect to the time windows, then so is $(R(E_1), r')$. The capacity constraints for each stack are not violated because items in different stacks for E_1 are also in different stacks for E_2, and items in the same stack for E_1 are also in the same stack for E_2, thus the capacity constraints are respected on $(R(E_1), r')$. The order in which the deliveries are performed on $(R(E_1), r')$ is the same as the order on $(R(E_2), r)$, i.e., each item that is in a stack for E_1 is also in the stack for E_2, and each item that has been ejected for E_1 can be ejected or not for E_2. No ejected item for E_2 can be in a stack for E_1. Thus, the relaxed multi-stack policy is not violated. Because $(R(E_2), r)$ is feasible, then so is $(R(E_1), r')$. Because the delivery triangle inequality holds for the reduced cost component c, the cost of r' does not exceed that of r. Thus, $c(E_1) \le c(E_2)$ implies that the cost of $(R(E_1), r')$ is at most equal to that of $(R(E_2), r)$. Hence, label E_1 dominates label E_2. $\qquad \square$
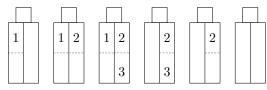
The reader can easily adapt this procedure to the non-elementary version of the shortest path problem.

4.2   Infeasible path cuts

When solving the shortest path problem with the labeling algorithm presented in the previous section, we might find a path for which the multi-stack policy is not respected. Figures 4a and 4b present a path and a

(a) Path
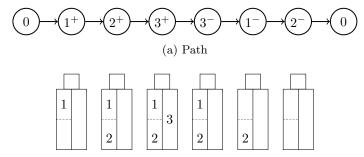
(b) Possible configuration of the vehicle not respecting the multi-stack policy

(c) Rearrangement of the items respecting the multi-stack policy

Figure 4: Path for which a possible rearrangement of the items can be found such that the multi-stack policy is respected, $q_1 = q_2 = q_3 = 1$ and $Q = 2$

(a) Path

(b) Possible configuration of the vehicle not respecting the multi-stack policy

Figure 5: Path for which no possible rearrangement of the items can be found such that the multi-stack policy is respected, $q_1 = q_2 = 1$, $q_3 = 2$ and $Q = 2$

configuration of the vehicle found with the relaxed multi-stack policy when setting $\kappa = 0$. This configuration does not respect the multi-stack policy, but the items can be rearranged as in Figure 4c in order to respect it. Thus, this path is feasible. Figures 5a and 5b illustrate a path and a configuration of the vehicle found with the relaxed multi-stack policy when setting $\kappa = 0$. In such a case, no rearrangement of the items is possible, and this path is infeasible.

In order to find out whether a path is feasible with respect to the multi-stack policy even if its current configuration is not, Côté et al. [13] proposed solving a bin packing problem. Instead, our algorithm solves a shortest path problem with multi-stack policy. The labeling algorithm presented in Section 3 is applied on the reduced graph containing only the arcs used in the current path. If a solution is found, a rearrangement is possible. Otherwise, the path is infeasible and its corresponding infeasible path inequality (20) is added to the RMP. Thus, these inequalities can be reformulated as

$$\sum_{r \in \Omega} \left(\sum_{\mu=1}^{\rho-1} b^r_{i_\mu, i_{\mu+1}} \right) y_r \leq |N(R)| - 2, \quad \forall R = (i_0, ..., i_\rho) \in \mathcal{R}^*. \tag{59}$$

These constraints are separated through an exact enumerative procedure. For every path in a given optimal solution of the master problem, several of these constraints can be violated. The sequential search is then

carried out on each active route $r \in \Omega$ with $y_r > 0$, and the first identified violated inequality is added to the master problem. Note that the dual variables of (59) affect the reduced cost of the arcs along the corresponding path, but we leave out the details for conciseness reasons.

5   Computational results

The two branch-price-and-cut algorithms just described were tested on a set of PDPTWMS instances derived from an instance of the TSPLIB. In this section, we report the computational results obtained for these PDPTWMS instances. The instances are solved by considering one, two, and three stacks. All tests were performed on a Linux computer equipped with an Intel(R) Core(TM) i7-3770 processor (3.4 GHz). The algorithms were implemented using the GENCOL library using CPLEX 12.4.0.0 to solve all restricted master problems.

5.1   Instances

To test our algorithms, we have generated 198 PDPTWMS instances from the *a280* instance of the TSPLIB by following the ideas of Carrabs et al. [8, 9], Cordeau et al. [12] for the TSPPDL, and of Côté et al. [13] for the PDTSPMS. Two classes of instances were tested. In the first class, C1, each item has a unit demand, and the total capacity of a vehicle is 6. In the second class, C2, the demand of each item is a random number between 3 and 9, and the capacity of a vehicle is 24 for the one- and two-stack variants, and 27 for the three-stack variant.

For each class, we have generated and tested a total of 99 instances in which the number of requests ranges from 25 to 75, i.e., the number of nodes ranges from 51 to 151. For an instance with $2n + 1$ nodes, we have kept the first $2n + 1$ nodes from the *a280* instance of the TSPLIB. For each request, a pickup and a delivery node have been randomly paired, and the time windows have been randomly generated. Three time window horizons were tested: (1) setting $\underline{w}_i \leq 500, \forall i \in P$ and $\underline{w}_i \leq 1000, \forall i \in D$, (2) setting $\underline{w}_i \leq 1000, \forall i \in P$ and $\underline{w}_i \leq 1200, \forall i \in D$, and (3) setting $\underline{w}_i \leq 1500, \forall i \in P$ and $\underline{w}_i \leq 2000, \forall i \in D$. The three time horizons are denoted by 500-1000, 1000-1200, and 1500-2000 in the following. For each time window horizon, three different time window lengths were tested, i.e., 15, 30, and 45.

In all instances, we first aim to minimize the number of vehicles. To this end, a fixed cost of 100,000 is imposed on each arc $(0, j) \in A$ with $j \in P$.

5.2   Detailed computational results

Table 1 presents the number of instances solved optimally for each algorithm using the elementary shortest path problem. A time limit of 7200 seconds (two hours) was imposed for the solution of each instance. The algorithms are denoted as follows: *BPC MS* (branch-price-and-cut algorithm with multi-stack feasible paths) and *BPC Relaxed* (branch-price-and-cut algorithm with relaxed multi-stack paths). For the latter, we also specify the value of κ. We have tested different values of κ and report those with $\kappa = 0$ and $\kappa = 2$.

For any number of stacks and instance class, *BPC MS* solves the largest number of instances. In our experiments, we observed that all instances solved with *BPC Relaxed* are also solved with *BPC MS*. Therefore,

Table 1: Number of instances solved for each algorithm

	BPC MS		BPC Relaxed $\kappa = 0$		BPC Relaxed $\kappa = 2$	
Class	C1	C2	C1	C2	C1	C2
1 stack	77	96	63	85	69	93
2 stacks	27	63	5	11	26	62
3 stacks	89	52	89	21	89	52

we will only present detailed computational results for the *BPC MS* algorithm. We have also tested an adaptation of the ng-path relaxation proposed by Baldacci et al. [3, 4] for the VRP and adapted by Cherkesly et al. [10] for the PDPTWL and the non-elementary shortest path problem. Neither of these two relaxations of the pricing problem has a positive impact on the quality of the lower bound or on the computational time. Thus, we do not present these results.

For the one- and two-stack variants, one can realize that the instances in class C1 are harder to solve than those in class C2. For the instances in class C1, this is probably due to the symmetry between the items, i.e., each item has unit demand. For the three-stack variant, the instances in class C2 prove to be more difficult. This is due to the increase of the maximum number of items simultaneously present in a vehicle. In fact, for each instance in class C1 solved with three stacks, the maximum number of items simultaneously in the vehicle is three (see Table A.10). Thus, solving the problem with the multi-stack policy for these instances is equivalent to solving the PDPTW.

Tables 2–4 present summarized results for the PDPTWMS with one, two, and three stacks, respectively, with the BPC MS algorithm when solving the elementary version of the suproblem in all three cases. Detailed computational results are presented in A. In each table we present, for each instance class, summarized results on each set of 11 instances with a specified time window length and horizon. For each of these 11 instances, the number of nodes ranges from 25 to 75. In each table, the first column indicates the width of the length of the time windows (15, 30 or 45), and the time window horizon, i.e., 500-1000, 1000-1200, and 1500-2000. For example, w15-500-1000 refers to instances that have a time window length of 15 and a time window horizon 500-1000. We present the following information: *NbSolved* the number of instances solved to optimality within the prescribed time limit; *Sec.*, the average CPU time in seconds; *Gap (%)*, the average integrity gap in percentage computed as $(z^* - \underline{z})/(z^*)$, where z^* is the optimal solution value and $\underline{z}$ is the lower bound at the root node before adding any cuts; *Veh*, the average minimal number of vehicles; and *maxItem*, the average maximal number of items simultaneously in a vehicle. All these averages are computed over the solved instances. Furthermore, for each class of instances, we report in the row *Weighted average* the averages for the class weighted according to the number of instances solved for each set of 11 instances with a specified time window length and horizon.

Table 2: Summarized computational results for the variant with one stack

Instance	NbSolved	Sec.	Gap (%)	Veh	maxItem
Instances in class C1					
w15-500-1000	11	59.9	0.00	24.18	3.64
w15-1000-1200	11	60.6	0.27	19.18	3.36
w15-1500-2000	7	98.6	0.39	12.71	4.14
w30-500-1000	10	1,783.1	0.53	20.00	3.80
w30-1000-1200	10	971.8	0.31	16.70	4.10
w30-1500-2000	8	654.3	0.00	12.63	4.25
w45-500-1000	7	2,130.4	1.92	15.00	4.57
w45-1000-1200	7	772.2	0.98	13.14	4.00
w45-1500-2000	6	785.1	0.69	11.17	4.17
Weighted average	8.56	777.0	0.50	16.86	3.95
Instances in class C2					
w15-500-1000	11	3.7	0.30	24.09	3.27
w15-1000-1200	11	3.8	0.25	19.18	3.45
w15-1500-2000	11	64.8	0.00	16.45	4.00
w30-500-1000	11	49.5	0.00	21.64	4.00
w30-1000-1200	11	21.5	0.57	18.00	3.64
w30-1500-2000	11	834.9	0.46	13.82	4.27
w45-500-1000	10	310.5	0.73	17.60	4.00
w45-1000-1200	11	267.3	1.24	15.27	3.73
w45-1500-2000	9	478.3	0.67	11.56	3.89
Weighted average	10.67	219.9	0.46	17.64	3.80

Table 3: Summarized computational results for the variant with two stacks

Instance	NbSolved	Sec.	Gap (%)	Veh	maxItem
Instances in class C1					
w15-500-1000	7	547.8	3.08	11.86	4.86
w15-1000-1200	5	2,036.2	1.87	9.80	4.20
w15-1500-2000	2	563.9	4.11	6.00	4.50
w30-500-1000	3	523.7	9.44	8.00	4.67
w30-1000-1200	3	2,396.4	4.80	8.33	5.00
w30-1500-2000	1	162.1	0.01	5.00	6.00
w45-500-1000	2	2,956.5	7.49	7.50	4.50
w45-1000-1200	3	3,480.1	10.95	6.33	5.33
w45-1500-2000	1	17.4	0.00	4.00	4.00
Weighted average	3.00	1,497.7	4.80	8.74	4.74
Instances in class C2					
w15-500-1000	11	541.4	2.75	14.91	4.27
w15-1000-1200	9	939.1	4.82	12.11	3.67
w15-1500-2000	6	1,391.0	5.58	7.83	4.00
w30-500-1000	6	124.0	3.16	10.83	4.50
w30-1000-1200	9	1,070.6	2.43	10.89	3.78
w30-1500-2000	6	1,410.6	2.91	7.00	4.00
w45-500-1000	4	419.7	5.89	8.25	4.25
w45-1000-1200	6	1,037.0	2.80	8.50	4.00
w45-1500-2000	6	1,234.6	3.39	7.17	4.50
Weighted average	7.00	903.2	3.59	10.35	4.08

Table 4: Summarized computational results for the variant with three stacks

Instance	NbSolved	Sec.	Gap (%)	Veh	maxItem
Instances in class C1					
w15-500-1000	11	294.3	3.29	16.00	3.00
w15-1000-1200	11	492.1	3.28	12.45	3.00
w15-1500-2000	10	844.8	4.62	10.80	3.00
w30-500-1000	11	246.6	2.55	15.64	3.00
w30-1000-1200	9	559.9	2.48	10.56	3.00
w30-1500-2000	10	1,584.8	1.71	10.40	3.00
w45-500-1000	11	543.2	1.94	15.09	3.00
w45-1000-1200	9	267.6	5.55	10.78	3.00
w45-1500-2000	7	1,071.2	2.29	8.57	3.00
Weighted average	9.89	635.7	3.07	12.53	3.00
Instances in class C2					
w15-500-1000	10	1,308.4	3.04	13.50	4.60
w15-1000-1200	8	726.5	4.94	11.25	4.25
w15-1500-2000	5	1,764.7	3.47	7.20	5.00
w30-500-1000	5	838.1	4.39	9.60	4.80
w30-1000-1200	7	825.2	3.67	9.43	4.00
w30-1500-2000	4	554.0	6.78	6.00	4.75
w45-500-1000	4	3,033.5	3.14	7.50	5.00
w45-1000-1200	5	287.3	3.68	7.40	4.20
w45-1500-2000	4	1,965.9	5.43	6.25	4.50
Weighted average	5.78	1,179.5	4.04	9.44	4.52

We first observe that for both classes of instances and independently of the number of stacks, solving instances with larger time windows and with larger time window horizons is harder. In fact, we can, solve 157, 135 and 112 instances with time window lengths of 15, 30 and 45, respectively, and we can solve 145, 145, and 114 instances with the time window horizons of 500-1000, 1000-1200, and 1500-2000, respectively.

Second, for instances with one and two stacks, instances of class C1 are harder to solve. For the one-stack variant, 77 instances in class C1 and 96 instances in class C2 are solved, and, for the two-stack variant, 27 instances in class C1 and 63 instances in class C2 are solved. This is probably due to the symmetry between the unit demand items. For instances solved with three stacks, we obtain the opposite result which is probably due to the increase of the number of items simultaneously in the vehicle for instances in class C2.

Third, solving instances in classes C1 and C2 with one stack yields better gaps than with two and three stacks. For class C1, the gaps are on average 0.50%, 4.80%, and 3.07% for the one-, two-, and three-stack variants, respectively. For class C2, the gaps are on average 0.46%, 3.59%, and 4.04% for the one-, two-, and three-stack variants, respectively. This is probably due to a lower number of feasible paths with one stack, which also explains why solving instances with one stack yields on average more vehicles in a solution. For class C1, the average number of vehicles is 16.86, 8.74, and 12.53, while for class C2, the average number of vehicles is 17.64, 10.35, and 9.44 for the one-, two-, and three-stack variants, respectively.

Fourth, for instances in class C1, solving them with three stacks yields better gaps than with two stacks, i.e., 3.07% and 4.80% on average, respectively. We can also observe that the average number of vehicles with three stacks is greater than with two stacks, i.e., 12.53 and 8.74 vehicles on average, respectively. This result holds for instances that are solved for both two and three stacks. Interestingly, not all feasible solutions with respect to two stacks are feasible with respect to three stacks because the stack capacities differ. In the proposed instances, each vehicle has a capacity of six: with two stacks, each stack has a capacity of three; and with three stacks, each stack has a capacity of two. Thus, from a managerial perspective, it is interesting to see that having more stacks in a vehicle does not necessarily reduce the total costs.

Finally, for instances in class C2, solving them with two stacks yields better gaps than with three stacks, i.e., 3.59% and 4.04% on average, but yields more vehicles, i.e., 10.35 and 9.44 vehicles on average, respectively. In the proposed instances, a vehicle with two stacks has a total capacity of 24 and a vehicle with three stacks has a total capacity of 27. Thus, an increase of 11.1% of the total vehicle capacity decreases, on average, the number of vehicles by 8.8% and the maximum number of items by 9.7% which seems coherent.

5.3   Impact of the number of stacks

Table 5 presents computational results showing the impact of the number of stacks on the total traveled distance, the number of vehicles used, and the maximum number of items simultaneously in a vehicle. Detailed computational results are presented in B. In the table, we compare the results obtained with one stack to the results obtained with two and three stacks. We present the following information: $\Delta\ Dist$ (%), the average relative difference in the total traveled distance, computed as $(Dist_2 - Dist_1)/(Dist_1)$ and $(Dist_3 - Dist_1)/(Dist_1)$, where $Dist_j, j = \{1, 2, 3\}$, is the distance with j stacks; $\Delta\ Veh$ (%) , the average relative difference in the number of vehicles, computed as $(Veh_2 - Veh_1)/(Veh_1)$ and $(Veh_3 - Veh_1)/(Veh_1)$, where $Veh_j, j = \{1, 2, 3\}$, is the number of vehicles with j stacks; and $\Delta\ maxItem$ (%), the average relative difference in the maximum number of items simultaneously in a vehicle, computed as $(maxItem_2 - maxItem_1)/(maxItem_1)$ and $(maxItem_3 - maxItem_1)/(maxItem_1)$, where $maxItem_j, j = \{1, 2, 3\}$, is the maximum number of items simultaneously in a vehicle with j stacks.

For instances in class C1, the total capacity of the vehicle is 6 independently of the number of stacks. First, increasing the number of stacks from one to two and from one to three decreases the total traveled distance by an average of 28.6% and 22.8%, and decreases the number of vehicles used by an average of 39.0% and 29.6%, respectively. Second, increasing the number of stacks from one to two increases the maximum number of items simultaneously in a vehicle by an average of 35.2%. Comparing one with three stacks, the maximum number of requests simultaneously in a vehicle decreases by an average of 21.6%. This last result is counterintuitive, but by examining the optimal solutions with three stacks, the maximum number of items is three and is often reached, whereas with one stack, this maximum is reached less often.

For the class C2, the total capacity of the vehicle is 24 for instances with one and two stacks, and 27 for instances with three stacks. Even though the capacity is not the same for the three-stack variant, we present the impact of the number of stacks on the results to show the general trend. We first observe that increasing the number of stacks from one to two and from one to three decreases the total traveled distance by an

Table 5: Summarized impact of the number of stacks on the results

Instance	1 stack VS 2 stacks			1 stack VS 3 stacks		
	Δ *Dist* (%)	Δ *Veh* (%)	Δ *maxItem* (%)	Δ *Dist* (%)	Δ *Veh* (%)	Δ *maxItem* (%)
Instances in class C1						
w15-500-1000	−32.0	−41.4	45.2	−28.3	−34.0	−15.0
w15-1000-1200	−25.9	−36.3	31.7	−23.3	−34.2	−9.1
w15-1500-2000	−31.7	−45.5	29.2	−22.9	−26.4	−25.7
w30-500-1000	−30.1	−40.1	30.6	−22.6	−26.3	−20.0
w30-1000-1200	−25.1	−34.3	38.9	−20.1	−33.4	−23.9
w30-1500-2000	−30.0	−37.5	50.0	−19.2	−26.6	−27.5
w45-500-1000	−27.1	−37.4	12.5	−17.9	−18.9	−33.6
w45-1000-1200	−27.0	−34.7	33.3	−24.2	−29.8	−25.0
w45-1500-2000	−23.9	−50.0	33.3	−25.1	−32.2	−25.8
Weighted average	−28.6	−39.0	35.2	−22.8	−29.6	−21.6
Instances in class C2						
w15-500-1000	−26.2	−37.6	32.6	−29.6	−40.4	45.0
w15-1000-1200	−23.2	−32.1	11.1	−26.2	−36.9	31.3
w15-1500-2000	−22.0	−38.9	11.1	−28.0	−42.0	41.7
w30-500-1000	−29.7	−42.2	20.8	−36.6	−49.0	30.0
w30-1000-1200	−18.5	−33.2	5.2	−22.2	−34.9	14.3
w30-1500-2000	−17.9	−36.0	5.6	−26.5	−40.2	31.3
w45-500-1000	−19.6	−30.4	14.6	−21.7	−37.1	33.3
w45-1000-1200	−21.1	−33.2	16.7	−26.4	−37.2	25.0
w45-1500-2000	−14.9	−30.3	18.1	−26.5	−35.2	22.9
Weighted average	−21.8	−35.0	15.8	−27.2	−39.1	31.4

average of 21.8% and 27.2%, respectively. It also decreases the number of vehicles used by an average of 35.0% and 39.1%, and increases the maximum number of requests simultaneously in a vehicle by an average of 15.8% and 31.4%, respectively.

Thus, for both classes C1 and C2, increasing the number of stacks from one to two has a positive impact on the total traveled distance and on the minimal number of vehicles needed. Interestingly, the additional gain of three stacks is significantly smaller than increasing the number of stacks from one to two.

6   Conclusions

In this paper, we have introduced the PDPTWMS and described two column generation algorithms to solve it. An ad hoc labeling algorithm for shortest path problems with multiple stacks is proposed and implemented. Moreover, we have adapted the hybrid branch-price-and-cut algorithm of Cherkesly et al. [10] for the PDPTWL to the PDPTWMS. In addition, we have introduced a new notation to represent a stack in a vehicle which can be adapted to variants of the PDPTW with loading constraints such as the PDPTW with handling costs. Instances involving up to 75 requests and three stacks were solved to optimality within two hours of computational time. On the PDPTWL instances, Cherkesly et al. [10] had shown that, for their instances, the *BPC Relaxed* seemed to outperform the *BPC MS*. On our new instances, we obtain the opposite result, i.e., the *BPC Relaxed* is outperformed by the *BPC MS*. Our results also show that increasing the number of stacks from one to two has a positive impact on the total traveled distance and on the minimal number of vehicles used, but increasing it from two to three does not yield a significant additional gain.

Appendix A   Detailed computational results

This appendix presents the detailed computational results on our test instances. Tables A.6–A.11 present the results obtained when solving each instance with the branch-price-and-cut algorithm with multi-stack feasible paths, and solving the elementary shortest path problem. In each table, the first column indicates the name of the instance corresponding to its number of nodes, its instance class (C1 or C2), the length of the time windows (15, 30 or 45), and the time window horizon. For example, instance a280-51-c1-w15-500-1000 involves 51 nodes, is in class C1, has a time window length of 15, and a time window horizon such that $\underline{w}_i \leq 500, \forall i \in P$ and $\underline{w}_i \leq 1000, \forall i \in D$. We present the following information: *Sec.*, the CPU time in seconds; $\underline{z}$, the lower bound at the root node before adding any cuts; z^*, the optimal solution value; *Veh.*, the minimal number of vehicles in the optimal solution; *maxItem*, the maximal number of items simultaneously in a vehicle; *OC*, the total number of constraints (46), (47), and (49) added to the master problem; and *B*, the number of nodes in the search tree including the root node. For all the instances tested with our algorithms, no feasible solution was found whenever the time limit was reached. Thus, we do not report a lower bound value even if one was found.

Table A.6: Computational results for the variant with one stack for instances in class C1

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c1-w15-500-1000	0.0	1,105,647.0	1,105,647.0	11	4	0	1
a280-61-c1-w15-500-1000	0.1	1,808,284.8	1,808,284.8	18	3	0	1
a280-71-c1-w15-500-1000	0.4	1,709,058.6	1,709,058.6	17	3	0	1
a280-81-c1-w15-500-1000	2.1	2,212,102.1	2,212,102.1	22	3	0	1
a280-91-c1-w15-500-1000	1.2	2,514,106.2	2,514,106.2	25	4	0	1
a280-101-c1-w15-500-1000	1.6	2,514,863.5	2,514,863.5	25	3	0	1
a280-111-c1-w15-500-1000	0.5	2,516,452.5	2,516,452.5	25	4	0	1
a280-121-c1-w15-500-1000	48.4	2,917,967.1	2,917,967.1	29	4	0	1
a280-131-c1-w15-500-1000	53.9	2,917,267.7	2,917,267.7	29	3	0	1
a280-141-c1-w15-500-1000	19.2	3,419,880.8	3,419,880.8	34	4	0	1
a280-151-c1-w15-500-1000	532.0	3,118,717.5	3,118,717.5	31	5	0	1
a280-51-c1-w15-1000-1200	0.1	1,105,937.2	1,105,937.2	11	3	0	1
a280-61-c1-w15-1000-1200	0.2	1,207,347.2	1,207,347.2	12	3	0	1
a280-71-c1-w15-1000-1200	0.5	1,308,216.2	1,308,216.2	13	3	0	1
a280-81-c1-w15-1000-1200	12.0	1,509,621.7	1,509,621.7	15	3	0	1
a280-91-c1-w15-1000-1200	8.2	2,112,198.8	2,112,198.8	21	4	0	1
a280-101-c1-w15-1000-1200	3.3	1,663,127.4	1,713,265.2	17	4	8	3
a280-111-c1-w15-1000-1200	6.4	2,113,948.0	2,113,948.0	21	3	0	1
a280-121-c1-w15-1000-1200	373.4	2,315,420.2	2,315,420.2	23	4	0	1
a280-131-c1-w15-1000-1200	119.8	2,114,163.8	2,114,163.8	21	4	0	1
a280-141-c1-w15-1000-1200	102.6	2,919,615.0	2,919,615.0	29	3	0	1
a280-151-c1-w15-1000-1200	40.0	2,817,812.9	2,817,812.9	28	3	0	1
a280-51-c1-w15-1500-2000	0.1	1,105,889.5	1,105,889.5	11	3	0	1
a280-61-c1-w15-1500-2000	1.7	1,106,952.5	1,106,952.5	11	4	0	1
a280-71-c1-w15-1500-2000	17.9	1,006,670.1	1,006,670.1	10	5	0	1
a280-81-c1-w15-1500-2000	6.2	1,008,842.1	1,008,842.1	10	5	0	1
a280-91-c1-w15-1500-2000	288.5	1,409,820.3	1,409,820.3	14	4	0	1
a280-101-c1-w15-1500-2000	68.6	1,510,936.0	1,510,936.0	15	4	0	1
a280-111-c1-w15-1500-2000	307.4	1,763,132.1	1,813,022.0	18	4	0	2
a280-121-c1-w15-1500-2000							
a280-131-c1-w15-1500-2000							
a280-141-c1-w15-1500-2000							
a280-151-c1-w15-1500-2000							
a280-51-c1-w30-500-1000	0.5	1,004,669.6	1,004,669.6	10	4	0	1
a280-61-c1-w30-500-1000	1.6	1,407,454.9	1,407,454.9	14	3	0	1
a280-71-c1-w30-500-1000	0.5	1,608,457.6	1,608,457.6	16	4	0	1
a280-81-c1-w30-500-1000	96.2	1,660,592.6	1,710,398.8	17	3	0	2
a280-91-c1-w30-500-1000	150.9	1,811,246.7	1,811,246.7	18	4	0	1
a280-101-c1-w30-500-1000	31.2	2,213,749.1	2,213,759.2	22	4	5	2
a280-111-c1-w30-500-1000	1,796.0	2,465,393.9	2,515,458.0	25	4	8	5
a280-121-c1-w30-500-1000	6,988.8	2,515,845.2	2,515,845.2	25	4	0	1
a280-131-c1-w30-500-1000	2,283.1	2,415,711.4	2,415,711.4	24	4	0	1
a280-141-c1-w30-500-1000							
a280-151-c1-w30-500-1000	6,481.7	2,906,085.9	2,918,603.3	29	4	2	3

Table A.6: Computational results for the variant with one stack for instances in class C1 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c1-w30-1000-1200	0.4	1,004,966.1	1,004,966.1	10	4	0	1
a280-61-c1-w30-1000-1200	0.5	1,206,358.7	1,206,358.7	12	3	0	1
a280-71-c1-w30-1000-1200	10.2	1,307,419.2	1,307,419.2	13	4	0	1
a280-81-c1-w30-1000-1200	20.1	1,409,119.5	1,409,119.5	14	4	0	1
a280-91-c1-w30-1000-1200	8.8	1,509,756.4	1,509,756.4	15	5	0	1
a280-101-c1-w30-1000-1200	22.1	1,562,241.1	1,612,187.1	16	4	0	6
a280-111-c1-w30-1000-1200	120.2	1,913,682.3	1,913,683.8	19	4	7	2
a280-121-c1-w30-1000-1200	1,647.7	2,215,871.3	2,215,871.3	22	4	0	1
a280-131-c1-w30-1000-1200							
a280-141-c1-w30-1000-1200	2,486.5	2,215,548.2	2,215,548.2	22	4	0	1
a280-151-c1-w30-1000-1200	5,401.1	2,415,924.4	2,415,924.4	24	5	0	1
a280-51-c1-w30-1500-2000	0.1	804,751.1	804,751.1	8	4	0	1
a280-61-c1-w30-1500-2000	6.5	805,473.4	805,473.4	8	4	0	1
a280-71-c1-w30-1500-2000	27.8	1,307,372.4	1,307,372.4	13	3	0	1
a280-81-c1-w30-1500-2000	16.6	1,208,314.2	1,208,314.2	12	4	0	1
a280-91-c1-w30-1500-2000	431.8	1,309,938.8	1,309,938.8	13	5	0	1
a280-101-c1-w30-1500-2000	412.2	1,411,245.3	1,411,245.3	14	4	0	1
a280-111-c1-w30-1500-2000	598.6	1,512,520.4	1,512,520.4	15	5	0	1
a280-121-c1-w30-1500-2000	3,740.8	1,813,294.2	1,813,294.2	18	5	0	1
a280-131-c1-w30-1500-2000							
a280-141-c1-w30-1500-2000							
a280-151-c1-w30-1500-2000							
a280-51-c1-w45-500-1000	1.7	1,105,693.2	1,105,693.2	11	4	0	1
a280-61-c1-w45-500-1000	9.9	1,256,820.3	1,306,849.4	13	4	0	2
a280-71-c1-w45-500-1000	269.2	1,073,686.5	1,107,077.6	11	5	14	6
a280-81-c1-w45-500-1000	1,382.7	1,459,158.4	1,509,170.9	15	4	12	3
a280-91-c1-w45-500-1000	803.1	1,609,893.8	1,609,893.8	16	5	0	1
a280-101-c1-w45-500-1000	6,504.3	1,946,106.6	2,012,805.5	20	5	0	2
a280-111-c1-w45-500-1000	5,941.8	1,911,894.5	1,911,894.5	19	5	0	1
a280-121-c1-w45-500-1000							
a280-131-c1-w45-500-1000							
a280-141-c1-w45-500-1000							
a280-151-c1-w45-500-1000							
a280-51-c1-w45-1000-1200	1.1	804,649.0	804,649.0	8	4	0	1
a280-61-c1-w45-1000-1200	1.6	905,443.6	905,443.6	9	4	0	1
a280-71-c1-w45-1000-1200	3.1	1,207,359.0	1,207,359.0	12	4	0	1
a280-81-c1-w45-1000-1200	3,041.7	1,358,979.4	1,409,060.4	14	4	0	62
a280-91-c1-w45-1000-1200	123.1	1,459,618.5	1,509,809.5	15	4	16	3
a280-101-c1-w45-1000-1200	1,751.2	1,711,792.9	1,711,812.7	17	4	13	12
a280-111-c1-w45-1000-1200	483.3	1,713,895.5	1,713,895.5	17	4	0	1
a280-121-c1-w45-1000-1200							
a280-131-c1-w45-1000-1200							
a280-141-c1-w45-1000-1200							
a280-151-c1-w45-1000-1200							
a280-51-c1-w45-1500-2000	0.1	805,142.4	805,142.4	8	3	0	1
a280-61-c1-w45-1500-2000	4.5	1,006,154.4	1,006,154.4	10	5	0	1
a280-71-c1-w45-1500-2000	36.3	1,007,200.9	1,007,200.9	10	5	0	1
a280-81-c1-w45-1500-2000	11.6	1,308,730.7	1,308,730.7	13	4	0	1
a280-91-c1-w45-1500-2000	50.7	1,159,978.6	1,209,905.5	12	4	0	2
a280-101-c1-w45-1500-2000	4,607.2	1,411,697.7	1,411,697.7	14	4	0	1
a280-111-c1-w45-1500-2000							
a280-121-c1-w45-1500-2000							
a280-131-c1-w45-1500-2000							
a280-141-c1-w45-1500-2000							
a280-151-c1-w45-1500-2000							

Table A.7: Computational results for the variant with one stack for instances in class C2

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c2-w15-500-1000	0.0	1,506,585.3	1,506,585.3	15	3	0	1
a280-61-c2-w15-500-1000	0.2	1,457,204.4	1,507,163.0	15	3	0	2
a280-71-c2-w15-500-1000	0.2	1,508,562.9	1,508,562.9	15	3	0	1
a280-81-c2-w15-500-1000	0.4	1,809,825.0	1,809,825.0	18	4	0	1
a280-91-c2-w15-500-1000	1.1	1,911,167.3	1,911,167.3	19	3	0	1
a280-101-c2-w15-500-1000	0.5	2,716,189.8	2,716,189.8	27	3	0	1
a280-111-c2-w15-500-1000	1.1	2,817,077.5	2,817,077.5	28	3	0	1
a280-121-c2-w15-500-1000	11.8	2,616,706.3	2,616,706.3	26	4	0	1
a280-131-c2-w15-500-1000	8.4	3,418,753.8	3,418,753.8	34	3	0	1
a280-141-c2-w15-500-1000	12.0	3,118,588.4	3,118,588.4	31	3	0	1
a280-151-c2-w15-500-1000	5.1	3,721,381.7	3,721,381.7	37	4	0	1
a280-51-c2-w15-1000-1200	0.0	1,206,090.1	1,206,090.1	12	3	0	1
a280-61-c2-w15-1000-1200	0.0	1,608,096.6	1,608,096.6	16	3	0	1
a280-71-c2-w15-1000-1200	0.1	1,407,622.1	1,407,622.1	14	3	0	1
a280-81-c2-w15-1000-1200	0.4	1,811,171.3	1,811,171.3	18	3	0	1
a280-91-c2-w15-1000-1200	0.5	1,811,570.2	1,811,570.2	18	4	0	1
a280-101-c2-w15-1000-1200	1.7	1,712,400.2	1,712,400.2	17	4	0	1
a280-111-c2-w15-1000-1200	2.4	1,913,406.5	1,913,406.5	19	3	0	1
a280-121-c2-w15-1000-1200	6.6	1,762,821.5	1,812,664.0	18	4	0	2
a280-131-c2-w15-1000-1200	6.9	2,215,077.3	2,215,077.3	22	4	0	1
a280-141-c2-w15-1000-1200	8.4	3,017,913.1	3,017,913.1	30	3	0	1
a280-151-c2-w15-1000-1200	15.3	2,718,035.8	2,718,035.8	27	4	0	1
a280-51-c2-w15-1500-2000	0.3	1,105,556.8	1,105,556.8	11	3	0	1
a280-61-c2-w15-1500-2000	0.1	1,006,239.9	1,006,239.9	10	4	0	1
a280-71-c2-w15-1500-2000	1.2	1,208,386.1	1,208,386.1	12	4	0	1
a280-81-c2-w15-1500-2000	2.9	1,609,497.7	1,609,497.7	16	3	0	1
a280-91-c2-w15-1500-2000	3.3	1,309,653.6	1,309,653.6	13	4	0	1
a280-101-c2-w15-1500-2000	7.3	1,511,317.0	1,511,317.0	15	4	0	1
a280-111-c2-w15-1500-2000	58.2	1,913,589.1	1,913,589.1	19	5	0	1
a280-121-c2-w15-1500-2000	57.2	1,613,194.9	1,613,194.9	16	4	0	1
a280-131-c2-w15-1500-2000	47.6	2,214,746.1	2,214,746.1	22	4	0	1
a280-141-c2-w15-1500-2000	75.7	2,415,844.1	2,415,844.1	24	5	0	1
a280-151-c2-w15-1500-2000	459.1	2,317,319.0	2,317,319.0	23	4	0	1
a280-51-c2-w30-500-1000	0.0	1,306,410.9	1,306,410.9	13	3	0	1
a280-61-c2-w30-500-1000	0.5	1,607,490.2	1,607,490.2	16	3	0	1
a280-71-c2-w30-500-1000	0.6	1,809,178.3	1,809,178.3	18	4	0	1
a280-81-c2-w30-500-1000	6.3	1,910,466.8	1,910,466.8	19	4	0	1
a280-91-c2-w30-500-1000	3.7	2,212,887.0	2,212,923.8	22	4	8	4
a280-101-c2-w30-500-1000	3.4	2,414,303.6	2,414,303.6	24	5	0	1
a280-111-c2-w30-500-1000	28.0	2,314,435.2	2,314,435.2	23	4	0	1
a280-121-c2-w30-500-1000	91.9	2,415,380.8	2,415,380.8	24	4	0	1
a280-131-c2-w30-500-1000	86.2	2,616,226.7	2,616,226.7	26	4	0	1
a280-141-c2-w30-500-1000	134.4	2,716,713.0	2,716,713.0	27	4	0	1
a280-151-c2-w30-500-1000	189.8	2,616,748.4	2,616,774.5	26	5	19	2
a280-51-c2-w30-1000-1200	0.0	904,649.6	904,649.6	9	3	0	1
a280-61-c2-w30-1000-1200	0.7	1,005,432.3	1,005,437.1	10	3	0	3
a280-71-c2-w30-1000-1200	1.0	1,407,500.7	1,407,507.1	14	5	4	2
a280-81-c2-w30-1000-1200	0.9	1,409,499.0	1,409,499.0	14	4	0	1
a280-91-c2-w30-1000-1200	1.0	1,811,139.4	1,811,139.4	18	3	0	1
a280-101-c2-w30-1000-1200	4.9	2,013,638.1	2,013,638.1	20	3	0	1
a280-111-c2-w30-1000-1200	51.1	1,863,134.4	1,913,243.5	19	4	14	28
a280-121-c2-w30-1000-1200	54.9	2,190,325.7	2,215,280.7	22	4	0	2
a280-131-c2-w30-1000-1200	16.3	1,963,883.7	2,013,900.9	20	4	0	2
a280-141-c2-w30-1000-1200	35.3	2,616,709.1	2,616,709.1	26	4	0	1
a280-151-c2-w30-1000-1200	70.6	2,616,560.0	2,616,560.0	26	3	0	1
a280-51-c2-w30-1500-2000	0.2	904,923.5	904,923.5	9	3	0	1
a280-61-c2-w30-1500-2000	0.5	706,464.3	706,464.3	7	4	0	1
a280-71-c2-w30-1500-2000	1.2	1,307,458.4	1,307,458.4	13	4	0	1
a280-81-c2-w30-1500-2000	2.1	1,208,839.3	1,208,839.3	12	4	0	1
a280-91-c2-w30-1500-2000	4.8	1,109,237.8	1,109,237.8	11	4	0	1
a280-101-c2-w30-1500-2000	58.2	1,245,157.2	1,311,314.2	13	5	0	2
a280-111-c2-w30-1500-2000	15.1	1,411,640.6	1,411,640.6	14	4	0	1
a280-121-c2-w30-1500-2000	118.0	1,512,718.8	1,512,718.8	15	5	0	1

Table A.7: Computational results for the variant with one stack for instances in class C2 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-131-c2-w30-1500-2000	326.5	1,713,721.4	1,713,721.4	17	5	0	1
a280-141-c2-w30-1500-2000	4,938.5	1,914,710.9	1,914,755.5	19	5	2	2
a280-151-c2-w30-1500-2000	3,719.1	2,216,229.3	2,216,235.2	22	4	9	4
a280-51-c2-w45-500-1000	0.5	1,005,077.4	1,005,077.4	10	3	0	1
a280-61-c2-w45-500-1000	20.2	1,005,400.6	1,005,434.9	10	4	20	5
a280-71-c2-w45-500-1000	16.3	1,307,698.0	1,307,744.3	13	4	14	6
a280-81-c2-w45-500-1000	3.6	1,509,460.8	1,509,460.8	15	4	0	1
a280-91-c2-w45-500-1000	8.3	1,409,459.2	1,409,459.2	14	4	0	1
a280-101-c2-w45-500-1000	163.9	1,807,855.2	1,912,082.1	19	4	43	4
a280-111-c2-w45-500-1000	105.6	1,913,493.3	1,913,493.3	19	4	0	1
a280-121-c2-w45-500-1000							
a280-131-c2-w45-500-1000	226.9	2,214,734.3	2,214,734.3	22	4	0	1
a280-141-c2-w45-500-1000	780.2	2,667,246.8	2,716,908.3	27	4	0	2
a280-151-c2-w45-500-1000	1,779.5	2,716,695.1	2,716,695.1	27	5	0	1
a280-51-c2-w45-1000-1200	0.1	1,005,255.3	1,005,255.3	10	3	0	1
a280-61-c2-w45-1000-1200	0.4	1,106,345.6	1,106,345.6	11	3	0	1
a280-71-c2-w45-1000-1200	1.6	1,206,790.8	1,206,858.9	12	4	6	2
a280-81-c2-w45-1000-1200	3.6	1,157,910.0	1,207,777.5	12	3	0	2
a280-91-c2-w45-1000-1200	2.9	1,410,414.7	1,410,414.7	14	4	0	1
a280-101-c2-w45-1000-1200	11.5	1,662,326.9	1,712,007.2	17	4	13	3
a280-111-c2-w45-1000-1200	16.7	1,612,507.0	1,612,507.0	16	4	0	1
a280-121-c2-w45-1000-1200	152.1	1,813,012.4	1,813,012.4	18	4	0	1
a280-131-c2-w45-1000-1200	745.3	1,813,883.3	1,913,259.6	19	4	25	17
a280-141-c2-w45-1000-1200	1,452.3	1,789,304.4	1,814,371.8	18	4	49	32
a280-151-c2-w45-1000-1200	553.3	2,116,420.0	2,116,463.9	21	4	12	5
a280-51-c2-w45-1500-2000	0.2	705,047.1	705,047.1	7	4	0	1
a280-61-c2-w45-1500-2000	0.5	1,106,182.1	1,106,182.1	11	3	0	1
a280-71-c2-w45-1500-2000	12.9	1,039,596.5	1,106,349.9	11	4	9	3
a280-81-c2-w45-1500-2000	20.1	908,183.5	908,466.8	9	4	9	9
a280-91-c2-w45-1500-2000	3.5	1,208,948.2	1,208,948.2	12	4	0	1
a280-101-c2-w45-1500-2000	23.9	1,211,215.2	1,211,215.2	12	4	0	1
a280-111-c2-w45-1500-2000	17.9	1,210,969.4	1,210,969.4	12	4	0	1
a280-121-c2-w45-1500-2000	368.5	1,413,167.6	1,413,207.2	14	4	4	5
a280-131-c2-w45-1500-2000	3,857.3	1,613,115.5	1,613,128.6	16	4	10	8
a280-141-c2-w45-1500-2000							
a280-151-c2-w45-1500-2000							

Table A.8: Computational results for the variant with two stacks for instances in class C1

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c1-w15-500-1000	1.8	704,140.1	704,140.1	7	5	0	1
a280-61-c1-w15-500-1000	51.5	955,662.7	1,005,105.4	10	5	0	2
a280-71-c1-w15-500-1000	74.3	1,014,253.7	1,106,783.1	11	4	13	4
a280-81-c1-w15-500-1000	867.4	1,207,975.7	1,207,975.7	12	6	0	1
a280-91-c1-w15-500-1000	1,677.1	1,309,472.0	1,309,504.1	13	4	3	8
a280-101-c1-w15-500-1000	693.5	1,359,661.3	1,409,613.4	14	5	0	2
a280-111-c1-w15-500-1000	469.2	1,535,073.1	1,611,220.5	16	5	17	3
a280-121-c1-w15-500-1000							
a280-131-c1-w15-500-1000							
a280-141-c1-w15-500-1000							
a280-151-c1-w15-500-1000							
a280-51-c1-w15-1000-1200	3.6	671,088.0	704,241.7	7	3	0	2
a280-61-c1-w15-1000-1200	27.1	805,276.5	805,276.5	8	5	0	1
a280-71-c1-w15-1000-1200	682.9	881,291.8	906,162.9	9	4	17	5
a280-81-c1-w15-1000-1200							
a280-91-c1-w15-1000-1200	4,935.1	1,209,115.3	1,209,115.3	12	5	0	1
a280-101-c1-w15-1000-1200							
a280-111-c1-w15-1000-1200	4,532.5	1,285,981.9	1,310,818.1	13	4	8	3
a280-121-c1-w15-1000-1200							
a280-131-c1-w15-1000-1200							
a280-141-c1-w15-1000-1200							
a280-151-c1-w15-1000-1200							

Table A.8: Computational results for the variant with two stacks for instances in class C1 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c1-w15-1500-2000	340.4	554,372.4	604,017.4	6	4	0	2
a280-61-c1-w15-1500-2000	787.4	604,751.2	604,751.2	6	5	0	1
a280-71-c1-w15-1500-2000							
a280-81-c1-w15-1500-2000							
a280-91-c1-w15-1500-2000							
a280-101-c1-w15-1500-2000							
a280-111-c1-w15-1500-2000							
a280-121-c1-w15-1500-2000							
a280-131-c1-w15-1500-2000							
a280-141-c1-w15-1500-2000							
a280-151-c1-w15-1500-2000							
a280-51-c1-w30-500-1000	711.5	553,767.5	603,547.1	6	5	7	3
a280-61-c1-w30-500-1000	494.6	714,272.0	804,773.3	8	5	21	3
a280-71-c1-w30-500-1000	365.0	917,222.5	1,005,906.0	10	4	25	7
a280-81-c1-w30-500-1000							
a280-91-c1-w30-500-1000							
a280-101-c1-w30-500-1000							
a280-111-c1-w30-500-1000							
a280-121-c1-w30-500-1000							
a280-131-c1-w30-500-1000							
a280-141-c1-w30-500-1000							
a280-151-c1-w30-500-1000							
a280-51-c1-w30-1000-1200	232.1	653,619.3	703,716.1	7	5	10	3
a280-61-c1-w30-1000-1200	165.6	704,727.8	704,727.8	7	5	0	1
a280-71-c1-w30-1000-1200							
a280-81-c1-w30-1000-1200							
a280-91-c1-w30-1000-1200							
a280-101-c1-w30-1000-1200	6,791.4	1,028,448.3	1,109,220.7	11	5	34	5
a280-111-c1-w30-1000-1200							
a280-121-c1-w30-1000-1200							
a280-131-c1-w30-1000-1200							
a280-141-c1-w30-1000-1200							
a280-151-c1-w30-1000-1200							
a280-51-c1-w30-1500-2000	162.1	503,285.3	503,325.0	5	6	39	16
a280-61-c1-w30-1500-2000							
a280-71-c1-w30-1500-2000							
a280-81-c1-w30-1500-2000							
a280-91-c1-w30-1500-2000							
a280-101-c1-w30-1500-2000							
a280-111-c1-w30-1500-2000							
a280-121-c1-w30-1500-2000							
a280-131-c1-w30-1500-2000							
a280-141-c1-w30-1500-2000							
a280-151-c1-w30-1500-2000							
a280-51-c1-w45-500-1000	2,845.0	644,466.8	704,103.9	7	4	27	21
a280-61-c1-w45-500-1000	3,068.0	752,580.7	805,048.0	8	5	2	3
a280-71-c1-w45-500-1000							
a280-81-c1-w45-500-1000							
a280-91-c1-w45-500-1000							
a280-101-c1-w45-500-1000							
a280-111-c1-w45-500-1000							
a280-121-c1-w45-500-1000							
a280-131-c1-w45-500-1000							
a280-141-c1-w45-500-1000							
a280-151-c1-w45-500-1000							
a280-51-c1-w45-1000-1200	1,198.3	472,853.8	503,405.3	5	5	5	3
a280-61-c1-w45-1000-1200	3,559.5	516,846.8	603,928.4	6	6	29	5
a280-71-c1-w45-1000-1200	5,682.5	705,751.0	805,415.2	8	5	55	17
a280-81-c1-w45-1000-1200							
a280-91-c1-w45-1000-1200							
a280-101-c1-w45-1000-1200							
a280-111-c1-w45-1000-1200							
a280-121-c1-w45-1000-1200							

Table A.8: Computational results for the variant with two stacks for instances in class C1 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-131-c1-w45-1000-1200							
a280-141-c1-w45-1000-1200							
a280-151-c1-w45-1000-1200							
a280-51-c1-w45-1500-2000	17.4	403,914.1	403,914.1	4	4	0	1
a280-61-c1-w45-1500-2000							
a280-71-c1-w45-1500-2000							
a280-81-c1-w45-1500-2000							
a280-91-c1-w45-1500-2000							
a280-101-c1-w45-1500-2000							
a280-111-c1-w45-1500-2000							
a280-121-c1-w45-1500-2000							
a280-131-c1-w45-1500-2000							
a280-141-c1-w45-1500-2000							
a280-151-c1-w45-1500-2000							

Table A.9: Computational results for the variant with two stacks for instances in class C2

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c2-w15-500-1000	6.1	804,375.0	804,506.1	8	4	48	16
a280-61-c2-w15-500-1000	1.9	867,904.7	905,317.9	9	4	0	2
a280-71-c2-w15-500-1000	6.0	956,421.7	1,006,320.0	10	4	0	2
a280-81-c2-w15-500-1000	8.7	1,250,523.1	1,307,681.1	13	5	15	3
a280-91-c2-w15-500-1000	110.3	1,258,642.3	1,308,469.4	13	5	26	11
a280-101-c2-w15-500-1000	54.0	1,611,601.6	1,611,817.0	16	4	25	13
a280-111-c2-w15-500-1000	69.9	1,787,934.1	1,812,796.4	18	4	12	4
a280-121-c2-w15-500-1000	2,738.1	1,626,114.1	1,712,384.1	17	4	53	39
a280-131-c2-w15-500-1000	2,177.2	1,797,801.6	1,813,249.3	18	5	36	37
a280-141-c2-w15-500-1000	551.6	1,964,742.4	2,014,119.1	20	4	35	10
a280-151-c2-w15-500-1000	231.2	2,142,837.7	2,215,602.5	22	4	20	4
a280-51-c2-w15-1000-1200	1.0	854,650.1	904,616.7	9	3	7	3
a280-61-c2-w15-1000-1200	1.4	822,315.4	905,305.4	9	3	0	2
a280-71-c2-w15-1000-1200	3.8	955,943.9	1,005,845.1	10	3	11	3
a280-81-c2-w15-1000-1200	41.1	1,174,609.8	1,208,065.7	12	4	23	11
a280-91-c2-w15-1000-1200	82.1	1,174,241.1	1,209,279.3	12	4	19	22
a280-101-c2-w15-1000-1200	563.8	1,222,364.3	1,310,248.8	13	4	64	39
a280-111-c2-w15-1000-1200	1,012.1	1,256,198.9	1,311,126.3	13	4	69	104
a280-121-c2-w15-1000-1200	669.2	1,138,498.8	1,209,965.6	12	4	54	9
a280-131-c2-w15-1000-1200							
a280-141-c2-w15-1000-1200	6,077.2	1,888,867.7	1,913,702.4	19	4	31	64
a280-151-c2-w15-1000-1200							
a280-51-c2-w15-1500-2000	8.1	554,299.1	603,995.5	6	4	31	4
a280-61-c2-w15-1500-2000	8.3	605,268.3	605,570.0	6	4	26	4
a280-71-c2-w15-1500-2000	153.5	726,483.4	805,775.4	8	4	27	9
a280-81-c2-w15-1500-2000	324.0	824,160.5	906,454.0	9	4	0	2
a280-91-c2-w15-1500-2000	749.4	865,179.4	908,308.7	9	4	61	18
a280-101-c2-w15-1500-2000	7,102.7	895,442.9	909,507.8	9	4	119	160
a280-111-c2-w15-1500-2000							
a280-121-c2-w15-1500-2000							
a280-131-c2-w15-1500-2000							
a280-141-c2-w15-1500-2000							
a280-151-c2-w15-1500-2000							
a280-51-c2-w30-500-1000	2.1	679,392.0	704,319.5	7	4	0	2
a280-61-c2-w30-500-1000	33.2	855,367.6	905,162.2	9	5	47	4
a280-71-c2-w30-500-1000	11.0	1,056,080.2	1,106,008.6	11	4	0	2
a280-81-c2-w30-500-1000	171.7	1,107,899.8	1,107,899.8	11	5	0	1
a280-91-c2-w30-500-1000	249.7	1,284,756.7	1,309,579.4	13	4	38	26
a280-101-c2-w30-500-1000	276.5	1,360,302.8	1,410,052.2	14	5	16	5
a280-111-c2-w30-500-1000							
a280-121-c2-w30-500-1000							
a280-131-c2-w30-500-1000							
a280-141-c2-w30-500-1000							
a280-151-c2-w30-500-1000							

Table A.9: Computational results for the variant with two stacks for instances in class C2 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c2-w30-1000-1200	0.5	704,314.8	704,324.1	7	3	3	2
a280-61-c2-w30-1000-1200	4.7	655,125.4	704,687.0	7	3	18	5
a280-71-c2-w30-1000-1200	12.5	906,049.3	906,094.7	9	4	9	2
a280-81-c2-w30-1000-1200	22.7	1,004,991.1	1,008,098.4	10	4	10	4
a280-91-c2-w30-1000-1200	129.7	1,168,697.1	1,208,357.5	12	4	19	15
a280-101-c2-w30-1000-1200	103.0	1,210,282.4	1,210,282.4	12	4	0	1
a280-111-c2-w30-1000-1200	4,133.4	1,276,467.0	1,311,048.2	13	3	107	221
a280-121-c2-w30-1000-1200							
a280-131-c2-w30-1000-1200	3,861.4	1,221,912.7	1,310,717.5	13	5	64	27
a280-141-c2-w30-1000-1200	1,367.8	1,485,128.7	1,512,917.1	15	4	5	3
a280-151-c2-w30-1000-1200							
a280-51-c2-w30-1500-2000	14.5	503,907.4	503,914.7	5	4	2	2
a280-61-c2-w30-1500-2000	11.9	505,504.5	505,504.5	5	4	0	1
a280-71-c2-w30-1500-2000	78.1	756,981.7	805,654.8	8	4	52	4
a280-81-c2-w30-1500-2000	84.9	700,065.3	707,568.9	7	4	30	4
a280-91-c2-w30-1500-2000	1,943.6	723,751.3	807,307.3	8	4	68	50
a280-101-c2-w30-1500-2000							
a280-111-c2-w30-1500-2000	6,330.7	909,872.6	910,183.0	9	4	75	43
a280-121-c2-w30-1500-2000							
a280-131-c2-w30-1500-2000							
a280-141-c2-w30-1500-2000							
a280-151-c2-w30-1500-2000							
a280-51-c2-w45-500-1000	29.2	644,309.0	703,927.4	7	4	11	5
a280-61-c2-w45-500-1000	495.7	721,863.2	804,639.2	8	4	20	4
a280-71-c2-w45-500-1000	83.8	806,421.5	806,421.5	8	5	0	1
a280-81-c2-w45-500-1000	1,070.1	958,832.5	1,007,186.1	10	4	62	43
a280-91-c2-w45-500-1000							
a280-101-c2-w45-500-1000							
a280-111-c2-w45-500-1000							
a280-121-c2-w45-500-1000							
a280-131-c2-w45-500-1000							
a280-141-c2-w45-500-1000							
a280-151-c2-w45-500-1000							
a280-51-c2-w45-1000-1200	0.9	703,894.7	703,894.7	7	4	0	1
a280-61-c2-w45-1000-1200	44.5	654,455.1	704,350.5	7	4	52	7
a280-71-c2-w45-1000-1200	27.7	705,805.6	705,809.2	7	4	1	2
a280-81-c2-w45-1000-1200	114.7	796,563.7	806,929.8	8	4	20	3
a280-91-c2-w45-1000-1200	427.1	983,156.6	1,007,973.5	10	4	54	15
a280-101-c2-w45-1000-1200	5,607.2	1,137,055.0	1,209,618.5	12	4	98	82
a280-111-c2-w45-1000-1200							
a280-121-c2-w45-1000-1200							
a280-131-c2-w45-1000-1200							
a280-141-c2-w45-1000-1200							
a280-151-c2-w45-1000-1200							
a280-51-c2-w45-1500-2000	53.0	479,238.6	504,155.2	5	5	61	22
a280-61-c2-w45-1500-2000	89.2	704,556.5	704,677.8	7	4	50	20
a280-71-c2-w45-1500-2000	1,644.7	655,713.5	705,136.8	7	5	62	10
a280-81-c2-w45-1500-2000	2,734.5	647,827.7	706,828.1	7	5	112	103
a280-91-c2-w45-1500-2000	293.3	809,319.0	809,325.4	8	4	3	2
a280-101-c2-w45-1500-2000							
a280-111-c2-w45-1500-2000	2,592.8	909,438.6	909,514.5	9	4	52	10
a280-121-c2-w45-1500-2000							
a280-131-c2-w45-1500-2000							
a280-141-c2-w45-1500-2000							
a280-151-c2-w45-1500-2000							

Table A.10: Computational results for the variant with three stacks for instances in class C1

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-51-c1-w15-500-1000	0.1	771,396.7	804,243.4	8	3	0	2
a280-61-c1-w15-500-1000	0.5	1,038,700.9	1,105,192.8	11	3	3	3
a280-71-c1-w15-500-1000	8.5	1,073,706.8	1,107,074.8	11	3	17	14
a280-81-c1-w15-500-1000	1.1	1,274,987.6	1,308,179.7	13	3	1	3
a280-91-c1-w15-500-1000	37.3	1,342,754.9	1,409,233.6	14	3	28	30
a280-101-c1-w15-500-1000	11.1	1,644,142.4	1,710,847.6	17	3	20	8
a280-111-c1-w15-500-1000	172.8	1,695,055.5	1,711,766.3	17	3	37	78
a280-121-c1-w15-500-1000	18.3	1,946,387.9	2,012,695.1	20	3	24	5
a280-131-c1-w15-500-1000	2,576.7	1,946,173.5	2,012,928.2	20	3	166	366
a280-141-c1-w15-500-1000	257.9	2,247,860.3	2,314,592.9	23	3	41	55
a280-151-c1-w15-500-1000	152.8	2,181,145.6	2,214,373.5	22	3	26	19
a280-51-c1-w15-1000-1200	4.2	637,638.0	704,178.9	7	3	20	28
a280-61-c1-w15-1000-1200	0.2	805,245.0	805,245.0	8	3	0	1
a280-71-c1-w15-1000-1200	0.6	892,260.1	906,384.6	9	3	0	2
a280-81-c1-w15-1000-1200	9.1	1,051,503.3	1,108,123.3	11	3	23	7
a280-91-c1-w15-1000-1200	19.1	1,242,724.8	1,309,120.9	13	3	27	10
a280-101-c1-w15-1000-1200	103.2	1,148,591.6	1,209,326.8	12	3	50	35
a280-111-c1-w15-1000-1200	238.6	1,344,351.1	1,411,124.3	14	3	60	46
a280-121-c1-w15-1000-1200	46.0	1,478,511.8	1,511,805.5	15	3	28	7
a280-131-c1-w15-1000-1200	1,719.5	1,396,817.3	1,412,334.7	14	3	80	158
a280-141-c1-w15-1000-1200	1,292.1	1,594,751.3	1,613,862.8	16	3	74	113
a280-151-c1-w15-1000-1200	1,980.0	1,801,830.6	1,814,289.0	18	3	42	202
a280-51-c1-w15-1500-2000	3.1	637,223.1	703,938.1	7	3	10	11
a280-61-c1-w15-1500-2000	21.8	738,422.2	805,122.9	8	3	24	36
a280-71-c1-w15-1500-2000	14.3	739,172.5	805,594.2	8	3	48	9
a280-81-c1-w15-1500-2000	41.5	807,266.7	807,350.6	8	3	54	15
a280-91-c1-w15-1500-2000	107.4	1,107,525.7	1,107,577.8	11	3	76	26
a280-101-c1-w15-1500-2000	424.0	1,042,218.4	1,108,513.7	11	3	91	104
a280-111-c1-w15-1500-2000	384.8	1,176,880.3	1,210,071.7	12	3	68	55
a280-121-c1-w15-1500-2000	1,917.6	1,343,986.4	1,410,583.7	14	3	81	122
a280-131-c1-w15-1500-2000	1,025.5	1,377,192.8	1,410,572.2	14	3	61	41
a280-141-c1-w15-1500-2000	4,507.6	1,444,926.8	1,511,435.5	15	3	101	231
a280-151-c1-w15-1500-2000							
a280-51-c1-w30-500-1000	7.1	703,832.1	703,905.1	7	3	17	7
a280-61-c1-w30-500-1000	8.3	871,468.7	904,887.6	9	3	47	10
a280-71-c1-w30-500-1000	0.7	1,013,737.7	1,106,203.5	11	3	0	2
a280-81-c1-w30-500-1000	1.1	1,274,842.3	1,308,059.0	13	3	0	2
a280-91-c1-w30-500-1000	69.4	1,342,490.8	1,409,180.7	14	3	40	24
a280-101-c1-w30-500-1000	14.8	1,577,926.0	1,611,012.6	16	3	23	5
a280-111-c1-w30-500-1000	66.3	1,677,678.4	1,710,981.0	17	3	41	14
a280-121-c1-w30-500-1000	1,701.3	1,945,921.7	2,012,740.7	20	3	172	236
a280-131-c1-w30-500-1000	27.2	2,012,655.2	2,012,673.4	20	3	6	3
a280-141-c1-w30-500-1000	245.0	2,280,698.1	2,313,981.9	23	3	23	16
a280-151-c1-w30-500-1000	571.0	2,214,890.5	2,214,964.9	22	3	65	38
a280-51-c1-w30-1000-1200	2.6	670,549.8	703,829.2	7	3	5	7
a280-61-c1-w30-1000-1200	1.6	705,245.5	705,282.5	7	3	10	2
a280-71-c1-w30-1000-1200	53.5	872,543.4	905,768.0	9	3	44	27
a280-81-c1-w30-1000-1200	13.0	907,210.3	907,261.1	9	3	10	5
a280-91-c1-w30-1000-1200	18.8	1,075,040.8	1,108,303.0	11	3	26	5
a280-101-c1-w30-1000-1200	75.7	1,065,388.4	1,109,516.5	11	3	54	9
a280-111-c1-w30-1000-1200	507.4	1,250,937.6	1,310,816.8	13	3	53	71
a280-121-c1-w30-1000-1200	1,542.5	1,378,738.1	1,411,791.0	14	3	69	106
a280-131-c1-w30-1000-1200							
a280-141-c1-w30-1000-1200	2,823.6	1,412,967.4	1,413,207.4	14	3	78	132
a280-151-c1-w30-1000-1200							
a280-51-c1-w30-1500-2000	0.3	504,074.5	504,074.5	5	3	0	1
a280-61-c1-w30-1500-2000	17.4	704,435.2	704,468.8	7	3	50	12
a280-71-c1-w30-1500-2000	169.2	772,317.7	805,558.0	8	3	78	51
a280-81-c1-w30-1500-2000	2.6	806,556.9	806,556.9	8	3	0	1
a280-91-c1-w30-1500-2000	87.6	1,041,509.9	1,108,169.4	11	3	31	15
a280-101-c1-w30-1500-2000	370.9	1,109,456.0	1,109,540.4	11	3	92	52
a280-111-c1-w30-1500-2000	1,543.6	1,109,733.2	1,109,871.9	11	3	96	156
a280-121-c1-w30-1500-2000	5,866.3	1,277,203.7	1,310,533.8	13	3	133	326

Table A.10: Computational results for the variant with three stacks for instances in class C1 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh.	maxItem	OC	B
a280-131-c1-w30-1500-2000	866.1	1,377,474.0	1,410,762.4	14	3	51	24
a280-141-c1-w30-1500-2000							
a280-151-c1-w30-1500-2000	6,923.6	1,579,566.0	1,612,873.2	16	3	76	205
a280-51-c1-w45-500-1000	3.9	670,998.4	704,064.9	7	3	7	5
a280-61-c1-w45-500-1000	0.5	905,467.5	905,467.5	9	3	0	1
a280-71-c1-w45-500-1000	7.4	972,894.5	1,006,087.9	10	3	10	4
a280-81-c1-w45-500-1000	793.0	1,274,267.2	1,307,675.7	13	3	211	201
a280-91-c1-w45-500-1000	175.0	1,408,377.3	1,408,428.5	14	3	63	29
a280-101-c1-w45-500-1000	257.2	1,576,915.6	1,610,308.2	16	3	44	54
a280-111-c1-w45-500-1000	439.2	1,677,230.6	1,710,508.8	17	3	41	32
a280-121-c1-w45-500-1000	360.3	1,744,599.4	1,811,360.8	18	3	47	18
a280-131-c1-w45-500-1000	55.6	1,912,004.9	1,912,031.6	19	3	25	4
a280-141-c1-w45-500-1000	3,685.7	2,080,515.1	2,113,707.4	21	3	150	195
a280-151-c1-w45-500-1000	197.0	2,181,064.3	2,214,374.5	22	3	32	9
a280-51-c1-w45-1000-1200	1.0	537,064.6	603,479.8	6	3	0	2
a280-61-c1-w45-1000-1200	11.9	616,449.2	704,077.5	7	3	42	5
a280-71-c1-w45-1000-1200	134.6	780,661.8	805,654.4	8	3	81	33
a280-81-c1-w45-1000-1200	332.8	906,496.2	906,578.9	9	3	61	44
a280-91-c1-w45-1000-1200	423.3	923,007.9	1,007,429.4	10	3	79	54
a280-101-c1-w45-1000-1200	566.1	1,142,730.9	1,209,398.5	12	3	50	60
a280-111-c1-w45-1000-1200							
a280-121-c1-w45-1000-1200	275.3	1,243,203.9	1,309,712.6	13	3	54	8
a280-131-c1-w45-1000-1200							
a280-141-c1-w45-1000-1200	110.3	1,444,918.3	1,511,335.9	15	3	5	3
a280-151-c1-w45-1000-1200	552.9	1,713,169.7	1,713,257.0	17	3	53	14
a280-51-c1-w45-1500-2000	0.7	403,943.5	403,943.5	4	3	0	1
a280-61-c1-w45-1500-2000	30.6	670,852.7	704,214.6	7	3	47	12
a280-71-c1-w45-1500-2000	33.1	805,772.5	805,819.7	8	3	43	8
a280-81-c1-w45-1500-2000	2,118.7	739,374.8	806,155.9	8	3	179	439
a280-91-c1-w45-1500-2000	74.9	807,624.2	807,681.5	8	3	57	7
a280-101-c1-w45-1500-2000	550.0	1,075,475.0	1,108,801.1	11	3	54	38
a280-111-c1-w45-1500-2000							
a280-121-c1-w45-1500-2000							
a280-131-c1-w45-1500-2000							
a280-141-c1-w45-1500-2000	4,690.5	1,411,266.7	1,411,337.1	14	3	64	103
a280-151-c1-w45-1500-2000							

Table A.11: Computational results for the variant with three stacks for instances in class C2

Instance	Sec.	$\underline{z}$	z^*	Veh	maxItem	OC	B
a280-51-c2-w15-500-1000	1.9	704,424.7	704,424.7	7	4	0	1
a280-61-c2-w15-500-1000	5.7	865,165.9	905,043.8	9	4	0	2
a280-71-c2-w15-500-1000	94.8	905,863.5	905,917.6	9	4	30	15
a280-81-c2-w15-500-1000	114.9	1,245,390.9	1,307,786.2	13	5	38	20
a280-91-c2-w15-500-1000	1,148.8	1,258,444.9	1,308,336.8	13	5	34	40
a280-101-c2-w15-500-1000	387.4	1,399,688.1	1,510,667.3	15	5	34	27
a280-111-c2-w15-500-1000	118.9	1,712,128.8	1,712,128.8	17	4	0	1
a280-121-c2-w15-500-1000	4,713.0	1,523,939.3	1,611,521.1	16	5	28	8
a280-131-c2-w15-500-1000	4,687.2	1,680,142.8	1,712,233.6	17	5	47	10
a280-141-c2-w15-500-1000	1,810.9	1,860,423.0	1,913,410.7	19	5	56	7
a280-151-c2-w15-500-1000							
a280-51-c2-w15-1000-1200	3.1	754,574.8	804,354.3	8	3	3	4
a280-61-c2-w15-1000-1200	3.9	771,728.7	804,876.3	8	4	0	2
a280-71-c2-w15-1000-1200	8.5	906,789.5	906,910.3	9	4	5	3
a280-81-c2-w15-1000-1200	33.4	1,132,702.8	1,207,431.8	12	4	10	4
a280-91-c2-w15-1000-1200	47.0	1,115,772.0	1,208,755.2	12	5	10	5
a280-101-c2-w15-1000-1200	994.2	1,134,718.4	1,209,308.8	12	5	28	18
a280-111-c2-w15-1000-1200	1,567.8	1,159,692.0	1,210,475.7	12	5	82	53
a280-121-c2-w15-1000-1200							
a280-131-c2-w15-1000-1200							
a280-141-c2-w15-1000-1200	3,154.4	1,708,632.4	1,713,062.9	17	4	12	3
a280-151-c2-w15-1000-1200							

Table A.11: Computational results for the variant with three stacks for instances in class C2 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh	maxItem	OC	B
a280-51-c2-w15-1500-2000	9.2	503,814.4	503,814.4	5	5	0	1
a280-61-c2-w15-1500-2000	8.9	585,037.5	604,889.9	6	5	0	2
a280-71-c2-w15-1500-2000	761.2	738,758.9	805,232.7	8	5	43	12
a280-81-c2-w15-1500-2000	6,018.1	873,171.3	906,400.1	9	5	19	7
a280-91-c2-w15-1500-2000	2,026.0	790,680.0	808,044.5	8	5	58	8
a280-101-c2-w15-1500-2000							
a280-111-c2-w15-1500-2000							
a280-121-c2-w15-1500-2000							
a280-131-c2-w15-1500-2000							
a280-141-c2-w15-1500-2000							
a280-151-c2-w15-1500-2000							
a280-51-c2-w30-500-1000	4.2	604,267.7	604,267.7	6	4	0	1
a280-61-c2-w30-500-1000	148.3	755,006.6	804,538.2	8	5	9	3
a280-71-c2-w30-500-1000	142.6	872,630.9	905,705.0	9	5	34	15
a280-81-c2-w30-500-1000							
a280-91-c2-w30-500-1000	299.1	1,129,789.1	1,208,022.4	12	5	29	10
a280-101-c2-w30-500-1000	3,596.1	1,234,939.8	1,309,393.0	13	5	64	16
a280-111-c2-w30-500-1000							
a280-121-c2-w30-500-1000							
a280-131-c2-w30-500-1000							
a280-141-c2-w30-500-1000							
a280-151-c2-w30-500-1000							
a280-51-c2-w30-1000-1200	0.7	704,314.8	704,324.1	7	3	3	2
a280-61-c2-w30-1000-1200	2.7	618,378.3	704,338.5	7	4	0	2
a280-71-c2-w30-1000-1200	262.3	791,007.5	806,080.0	8	5	51	28
a280-81-c2-w30-1000-1200	61.3	969,668.5	1,006,878.1	10	4	14	4
a280-91-c2-w30-1000-1200	557.3	1,051,651.3	1,107,908.5	11	4	43	20
a280-101-c2-w30-1000-1200	1,165.2	1,077,967.4	1,109,227.0	11	4	33	7
a280-111-c2-w30-1000-1200	3,727.0	1,210,422.9	1,210,549.7	12	4	43	41
a280-121-c2-w30-1000-1200							
a280-131-c2-w30-1000-1200							
a280-141-c2-w30-1000-1200							
a280-151-c2-w30-1000-1200							
a280-51-c2-w30-1500-2000	82.3	437,036.0	503,573.7	5	6	14	4
a280-61-c2-w30-1500-2000	105.5	504,662.3	504,708.6	5	4	40	3
a280-71-c2-w30-1500-2000	600.0	672,606.3	705,468.4	7	4	64	13
a280-81-c2-w30-1500-2000	1,428.1	641,417.2	706,655.6	7	5	52	20
a280-91-c2-w30-1500-2000							
a280-101-c2-w30-1500-2000							
a280-111-c2-w30-1500-2000							
a280-121-c2-w30-1500-2000							
a280-131-c2-w30-1500-2000							
a280-141-c2-w30-1500-2000							
a280-151-c2-w30-1500-2000							
a280-51-c2-w45-500-1000	39.4	604,419.3	604,419.3	6	4	0	1
a280-61-c2-w45-500-1000	2,702.7	677,875.0	704,438.8	7	5	40	4
a280-71-c2-w45-500-1000	3,554.5	752,402.1	805,445.0	8	6	44	17
a280-81-c2-w45-500-1000	5,837.3	887,015.6	907,002.7	9	5	62	81
a280-91-c2-w45-500-1000							
a280-101-c2-w45-500-1000							
a280-111-c2-w45-500-1000							
a280-121-c2-w45-500-1000							
a280-131-c2-w45-500-1000							
a280-141-c2-w45-500-1000							
a280-151-c2-w45-500-1000							
a280-51-c2-w45-1000-1200	5.4	703,866.0	703,880.1	7	4	10	2
a280-61-c2-w45-1000-1200	34.5	554,048.8	603,773.1	6	4	0	2
a280-71-c2-w45-1000-1200	86.3	671,978.4	705,150.7	7	5	21	4
a280-81-c2-w45-1000-1200	837.1	777,024.2	806,567.4	8	4	48	11
a280-91-c2-w45-1000-1200	473.1	891,572.5	907,833.8	9	4	51	6
a280-101-c2-w45-1000-1200							
a280-111-c2-w45-1000-1200							
a280-121-c2-w45-1000-1200							

Table A.11: Computational results for the variant with three stacks for instances in class C2 (continued)

Instance	Sec.	$\underline{z}$	z^*	Veh	maxItem	OC	B
a280-131-c2-w45-1000-1200							
a280-141-c2-w45-1000-1200							
a280-151-c2-w45-1000-1200							
a280-51-c2-w45-1500-2000	20.3	470,300.9	503,496.4	5	4	0	2
a280-61-c2-w45-1500-2000	48.6	537,642.8	604,088.8	6	5	32	4
a280-71-c2-w45-1500-2000							
a280-81-c2-w45-1500-2000	3,483.0	606,788.8	607,006.4	6	5	50	20
a280-91-c2-w45-1500-2000	4,311.8	773,673.9	806,792.8	8	4	61	8
a280-101-c2-w45-1500-2000							
a280-111-c2-w45-1500-2000							
a280-121-c2-w45-1500-2000							
a280-131-c2-w45-1500-2000							
a280-141-c2-w45-1500-2000							
a280-151-c2-w45-1500-2000							

Appendix B   Detailed results on the impact of the number of stacks

This appendix presents detailed computational results to assess the impact of the number of stacks. Tables B.12 and B.13 present the impact of the number of stacks on the computational results for instances in class C1 and C2, respectively. For each table, we compare the results obtained with one stack to the results obtained with two and three stacks. We present the following information: Δ *Dist (%)*, the impact in percentage on the total traveled distance, computed as $(Dist_2 - Dist_1)/(Dist_1)$ and $(Dist_3 - Dist_1)/(Dist_1)$, where $Dist_j, j = \{1, 2, 3\}$, is the distance with j stacks, respectively; Δ *Veh (%)*, the impact in percentage on the number of vehicles, computed as $(Veh_2 - Veh_1)/(Veh_1)$ and $(Veh_3 - Veh_1)/(Veh_1)$, where $Veh_j, j = \{1, 2, 3\}$, is the number of vehicles with j stacks, respectively; and Δ *maxItem (%)*, the impact in percentage on the maximum number of items simultaneously in a vehicle, computed as $(maxItem_2 - maxItem_1)/(maxItem_1)$ and $(maxItem_3 - maxItem_1)/(maxItem_1)$, where $maxItem_j, j = \{1, 2, 3\}$, is the maximum number of items simultaneously in a vehicle with j stacks.

Table B.12: Impact of the number of stacks on the results for instances in class C1

Instance	1 stack VS 2 stacks			1 stack VS 3 stacks		
	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)
a280-51-c1-w15-500-1000	−26.7	−36.4	25.0	−24.9	−27.3	−25.0
a280-61-c1-w15-500-1000	−38.4	−44.4	66.7	−37.3	−38.9	0.0
a280-71-c1-w15-500-1000	−25.1	−35.3	33.3	−21.9	−35.3	0.0
a280-81-c1-w15-500-1000	−34.1	−45.5	100.0	−32.4	−40.9	0.0
a280-91-c1-w15-500-1000	−32.6	−48.0	0.0	−34.5	−44.0	−25.0
a280-101-c1-w15-500-1000	−35.3	−44.0	66.7	−27.0	−32.0	0.0
a280-111-c1-w15-500-1000	−31.8	−36.0	25.0	−28.5	−32.0	−25.0
a280-121-c1-w15-500-1000				−29.3	−31.0	−25.0
a280-131-c1-w15-500-1000				−25.1	−31.0	0.0
a280-141-c1-w15-500-1000				−26.6	−32.4	−25.0
a280-151-c1-w15-500-1000				−23.2	−29.0	−40.0
a280-51-c1-w15-1000-1200	−28.6	−36.4	0.0	−29.6	−36.4	0.0
a280-61-c1-w15-1000-1200	−28.2	−33.3	66.7	−28.6	−33.3	0.0
a280-71-c1-w15-1000-1200	−25.0	−30.8	33.3	−22.3	−30.8	0.0
a280-81-c1-w15-1000-1200				−15.6	−26.7	0.0
a280-91-c1-w15-1000-1200	−25.3	−42.9	25.0	−25.2	−38.1	−25.0
a280-101-c1-w15-1000-1200				−29.7	−29.4	−25.0
a280-111-c1-w15-1000-1200	−22.4	−38.1	33.3	−20.2	−33.3	0.0
a280-121-c1-w15-1000-1200				−23.4	−34.8	−25.0
a280-131-c1-w15-1000-1200				−12.9	−33.3	−25.0
a280-141-c1-w15-1000-1200				−29.3	−44.8	0.0
a280-151-c1-w15-1000-1200				−19.8	−35.7	0.0
a280-51-c1-w15-1500-2000	−31.8	−45.5	33.3	−33.1	−36.4	0.0
a280-61-c1-w15-1500-2000	−31.7	−45.5	25.0	−26.3	−27.3	−25.0

Table B.12: Impact of the number of stacks on the results for instances in class C1 (continued)

Instance	1 stack VS 2 stacks			1 stack VS 3 stacks		
	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)
a280-71-c1-w15-1500-2000				−16.1	−20.0	−40.0
a280-81-c1-w15-1500-2000				−16.9	−20.0	−40.0
a280-91-c1-w15-1500-2000				−22.8	−21.4	−25.0
a280-101-c1-w15-1500-2000				−22.1	−26.7	−25.0
a280-111-c1-w15-1500-2000				−22.7	−33.3	−25.0
a280-121-c1-w15-1500-2000						
a280-131-c1-w15-1500-2000						
a280-141-c1-w15-1500-2000						
a280-151-c1-w15-1500-2000						
a280-51-c1-w30-500-1000	−24.0	−40.0	25.0	−16.4	−30.0	−25.0
a280-61-c1-w30-500-1000	−36.0	−42.9	66.7	−34.4	−35.7	0.0
a280-71-c1-w30-500-1000	−30.2	−37.5	0.0	−26.7	−31.3	−25.0
a280-81-c1-w30-500-1000				−22.5	−23.5	0.0
a280-91-c1-w30-500-1000				−18.4	−22.2	−25.0
a280-101-c1-w30-500-1000				−20.0	−27.3	−25.0
a280-111-c1-w30-500-1000				−29.0	−32.0	−25.0
a280-121-c1-w30-500-1000				−19.6	−20.0	−25.0
a280-131-c1-w30-500-1000				−19.3	−16.7	−25.0
a280-141-c1-w30-500-1000						
a280-151-c1-w30-500-1000				−19.6	−24.1	−25.0
a280-51-c1-w30-1000-1200	−25.2	−30.0	25.0	−22.9	−30.0	−25.0
a280-61-c1-w30-1000-1200	−25.6	−41.7	66.7	−16.9	−41.7	0.0
a280-71-c1-w30-1000-1200				−22.3	−30.8	−25.0
a280-81-c1-w30-1000-1200				−20.4	−35.7	−25.0
a280-91-c1-w30-1000-1200				−14.9	−26.7	−40.0
a280-101-c1-w30-1000-1200	−24.3	−31.3	25.0	−21.9	−31.3	-25.0
a280-111-c1-w30-1000-1200				−21.0	−31.6	−25.0
a280-121-c1-w30-1000-1200				−25.7	−36.4	−25.0
a280-131-c1-w30-1000-1200						
a280-141-c1-w30-1000-1200				−15.1	−36.4	−25.0
a280-151-c1-w30-1000-1200						
a280-51-c1-w30-1500-2000	−30.0	−37.5	50.0	−14.2	−37.5	−25.0
a280-61-c1-w30-1500-2000				−18.4	−12.5	−25.0
a280-71-c1-w30-1500-2000				−24.6	−38.5	0.0
a280-81-c1-w30-1500-2000				−21.1	−33.3	−25.0
a280-91-c1-w30-1500-2000				−17.8	−15.4	−40.0
a280-101-c1-w30-1500-2000				−15.2	−21.4	−25.0
a280-111-c1-w30-1500-2000				−21.2	−26.7	−40.0
a280-121-c1-w30-1500-2000				−20.8	−27.8	−40.0
a280-131-c1-w30-1500-2000						
a280-141-c1-w30-1500-2000						
a280-151-c1-w30-1500-2000						
a280-51-c1-w45-500-1000	−27.9	−36.4	0.0	−28.6	−36.4	−25.0
a280-61-c1-w45-500-1000	−26.3	−38.5	25.0	−20.2	−30.8	−25.0
a280-71-c1-w45-500-1000				−14.0	−9.1	−40.0
a280-81-c1-w45-500-1000				−16.3	−13.3	−25.0
a280-91-c1-w45-500-1000				−14.8	−12.5	−40.0
a280-101-c1-w45-500-1000				−19.5	−20.0	−40.0
a280-111-c1-w45-500-1000				−11.6	−10.5	−40.0
a280-121-c1-w45-500-1000						
a280-131-c1-w45-500-1000						
a280-141-c1-w45-500-1000						
a280-151-c1-w45-500-1000						
a280-51-c1-w45-1000-1200	−26.8	−37.5	25.0	−25.2	−25.0	−25.0
a280-61-c1-w45-1000-1200	−27.8	−33.3	50.0	−25.1	−22.2	−25.0
a280-71-c1-w45-1000-1200	−26.4	−33.3	25.0	−23.2	−33.3	−25.0
a280-81-c1-w45-1000-1200				−27.4	−35.7	−25.0
a280-91-c1-w45-1000-1200				−24.3	−33.3	−25.0
a280-101-c1-w45-1000-1200				−20.4	−29.4	−25.0
a280-111-c1-w45-1000-1200						
a280-121-c1-w45-1000-1200						
a280-131-c1-w45-1000-1200						

Table B.12: Impact of the number of stacks on the results for instances in class C1 (continued)

	1 stack VS 2 stacks			1 stack VS 3 stacks		
Instance	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)
a280-141-c1-w45-1000-1200						
a280-151-c1-w45-1000-1200						
a280-51-c1-w45-1500-2000	−23.9	−50.0	33.3	−23.3	−50.0	0.0
a280-61-c1-w45-1500-2000				−31.5	−30.0	−40.0
a280-71-c1-w45-1500-2000				−19.2	−20.0	−40.0
a280-81-c1-w45-1500-2000				−29.5	−38.5	−25.0
a280-91-c1-w45-1500-2000				−22.5	−33.3	−25.0
a280-101-c1-w45-1500-2000				−24.8	−21.4	−25.0
a280-111-c1-w45-1500-2000						
a280-121-c1-w45-1500-2000						
a280-131-c1-w45-1500-2000						
a280-141-c1-w45-1500-2000						
a280-151-c1-w45-1500-2000						
Average	−28.6	−39.0	35.2	−22.8	−29.6	−21.6

Table B.13: Impact of the number of stacks on the results for instances in class C2

	1 stack VS 2 stacks			1 stack VS 3 stacks		
Instance	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)
a280-51-c2-w15-500-1000	−31.6	−46.7	33.3	−32.8	−53.3	33.3
a280-61-c2-w15-500-1000	−25.8	−40.0	33.3	−29.6	−40.0	33.3
a280-71-c2-w15-500-1000	−26.2	−33.3	33.3	−30.9	−40.0	33.3
a280-81-c2-w15-500-1000	−21.8	−27.8	25.0	−20.8	−27.8	25.0
a280-91-c2-w15-500-1000	−24.2	−31.6	66.7	−25.3	−31.6	66.7
a280-101-c2-w15-500-1000	−27.0	−40.7	33.3	−34.1	−44.4	66.7
a280-111-c2-w15-500-1000	−25.1	−35.7	33.3	−29.0	−39.3	33.3
a280-121-c2-w15-500-1000	−25.9	−34.6	0.0	−31.0	−38.5	25.0
a280-131-c2-w15-500-1000	−29.4	−47.1	66.7	−34.8	−50.0	66.7
a280-141-c2-w15-500-1000	−24.0	−35.5	33.3	−27.9	−38.7	66.7
a280-151-c2-w15-500-1000	−27.0	−40.5	0.0			
a280-51-c2-w15-1000-1200	−24.2	−25.0	0.0	−28.5	−33.3	0.0
a280-61-c2-w15-1000-1200	−34.5	−43.8	0.0	−39.8	−50.0	33.3
a280-71-c2-w15-1000-1200	−23.3	−28.6	0.0	−9.3	−35.7	33.3
a280-81-c2-w15-1000-1200	−27.8	−33.3	33.3	−33.5	−33.3	33.3
a280-91-c2-w15-1000-1200	−19.8	−33.3	0.0	−24.3	−33.3	25.0
a280-101-c2-w15-1000-1200	−17.4	−23.5	0.0	−24.9	−29.4	25.0
a280-111-c2-w15-1000-1200	−17.0	−31.6	33.3	−21.9	−36.8	66.7
a280-121-c2-w15-1000-1200	−21.3	−33.3	0.0			
a280-131-c2-w15-1000-1200						
a280-141-c2-w15-1000-1200	−23.5	−36.7	33.3	−27.1	−43.3	33.3
a280-151-c2-w15-1000-1200						
a280-51-c2-w15-1500-2000	−28.1	−45.5	33.3	−31.4	−54.5	66.7
a280-61-c2-w15-1500-2000	−10.7	−40.0	0.0	−21.6	−40.0	25.0
a280-71-c2-w15-1500-2000	−31.1	−33.3	0.0	−37.6	−33.3	25.0
a280-81-c2-w15-1500-2000	−32.0	−43.8	33.3	−32.6	−43.8	66.7
a280-91-c2-w15-1500-2000	−13.9	−30.8	0.0	−16.7	−38.5	25.0
a280-101-c2-w15-1500-2000	−16.0	−40.0	0.0			
a280-111-c2-w15-1500-2000						
a280-121-c2-w15-1500-2000						
a280-131-c2-w15-1500-2000						
a280-141-c2-w15-1500-2000						
a280-151-c2-w15-1500-2000						
a280-51-c2-w30-500-1000	−32.6	−46.2	33.3	−33.4	−53.8	33.3
a280-61-c2-w30-500-1000	−31.1	−43.8	66.7	−39.4	−50.0	66.7
a280-71-c2-w30-500-1000	−34.5	−38.9	0.0	−37.8	−50.0	25.0
a280-81-c2-w30-500-1000	−24.5	−42.1	25.0			
a280-91-c2-w30-500-1000	−25.9	−40.9	0.0	−37.9	−45.5	25.0
a280-101-c2-w30-500-1000	−29.7	−41.7	0.0	−34.3	−45.8	0.0
a280-111-c2-w30-500-1000						

Table B.13: Impact of the number of stacks on the results for instances in class C2 (continued)

Instance	1 stack VS 2 stacks			1 stack VS 3 stacks		
	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)	Δ Dist (%)	Δ Veh (%)	Δ maxItem (%)
a280-121-c2-w30-500-1000						
a280-131-c2-w30-500-1000						
a280-141-c2-w30-500-1000						
a280-151-c2-w30-500-1000						
a280-51-c2-w30-1000-1200	−7.0	−22.2	0.0	−7.0	−22.2	0.0
a280-61-c2-w30-1000-1200	−13.8	−30.0	0.0	−20.2	−30.0	33.3
a280-71-c2-w30-1000-1200	−18.8	−35.7	−20.0	−19.0	−42.9	0.0
a280-81-c2-w30-1000-1200	−14.7	−28.6	0.0	−27.6	−28.6	0.0
a280-91-c2-w30-1000-1200	−25.0	−33.3	33.3	−29.0	−38.9	33.3
a280-101-c2-w30-1000-1200	−24.6	−40.0	33.3	−32.3	−45.0	33.3
a280-111-c2-w30-1000-1200	−16.6	−31.6	−25.0	−20.3	−36.8	0.0
a280-121-c2-w30-1000-1200						
a280-131-c2-w30-1000-1200	−22.9	−35.0	25.0			
a280-141-c2-w30-1000-1200	−22.7	−42.3	0.0			
a280-151-c2-w30-1000-1200						
a280-51-c2-w30-1500-2000	−20.5	−44.4	33.3	−27.4	−44.4	100.0
a280-61-c2-w30-1500-2000	−14.8	−28.6	0.0	−27.2	−28.6	0.0
a280-71-c2-w30-1500-2000	−24.2	−38.5	0.0	−26.7	−46.2	0.0
a280-81-c2-w30-1500-2000	−14.4	−41.7	0.0	−24.7	−41.7	25.0
a280-91-c2-w30-1500-2000	−20.9	−27.3	0.0			
a280-101-c2-w30-1500-2000						
a280-111-c2-w30-1500-2000	−12.5	−35.7	0.0			
a280-121-c2-w30-1500-2000						
a280-131-c2-w30-1500-2000						
a280-141-c2-w30-1500-2000						
a280-151-c2-w30-1500-2000						
a280-51-c2-w45-500-1000	−22.6	−30.0	33.3	−13.0	−40.0	33.3
a280-61-c2-w45-500-1000	−14.6	−20.0	0.0	−18.3	−30.0	25.0
a280-71-c2-w45-500-1000	−17.1	−38.5	25.0	−29.7	−38.5	50.0
a280-81-c2-w45-500-1000	−24.0	−33.3	0.0	−26.0	−40.0	25.0
a280-91-c2-w45-500-1000						
a280-101-c2-w45-500-1000						
a280-111-c2-w45-500-1000						
a280-121-c2-w45-500-1000						
a280-131-c2-w45-500-1000						
a280-141-c2-w45-500-1000						
a280-151-c2-w45-500-1000						
a280-51-c2-w45-1000-1200	−25.9	−30.0	33.3	−26.2	−30.0	33.3
a280-61-c2-w45-1000-1200	−31.4	−36.4	33.3	−40.5	−45.5	33.3
a280-71-c2-w45-1000-1200	−15.3	−41.7	0.0	−24.9	−41.7	25.0
a280-81-c2-w45-1000-1200	−10.9	−33.3	33.3	−15.6	−33.3	33.3
a280-91-c2-w45-1000-1200	−23.4	−28.6	0.0	−24.8	−35.7	0.0
a280-101-c2-w45-1000-1200	−19.9	−29.4	0.0			
a280-111-c2-w45-1000-1200						
a280-121-c2-w45-1000-1200						
a280-131-c2-w45-1000-1200						
a280-141-c2-w45-1000-1200						
a280-151-c2-w45-1000-1200						
a280-51-c2-w45-1500-2000	−17.7	−28.6	25.0	−30.7	−28.6	0.0
a280-61-c2-w45-1500-2000	−24.3	−36.4	33.3	−33.9	−45.5	66.7
a280-71-c2-w45-1500-2000	−19.1	−36.4	25.0			
a280-81-c2-w45-1500-2000	−19.4	−22.2	25.0	−17.2	−33.3	25.0
a280-91-c2-w45-1500-2000	4.2	−33.3	0.0	−24.1	−33.3	0.0
a280-101-c2-w45-1500-2000						
a280-111-c2-w45-1500-2000	−13.3	−25.0	0.0			
a280-121-c2-w45-1500-2000						
a280-131-c2-w45-1500-2000						
a280-141-c2-w45-1500-2000						
a280-151-c2-w45-1500-2000						
Average	−21.8	−35.0	15.8	−27.2	−39.1	31.4

References

[1] Alba Martínez, M., Cordeau, J.-F., Dell'Amico, M., Iori, M., 2013. A branch-and-cut algorithm for the double traveling salesman problem with multiple stacks. INFORMS Journal on Computing 25(1), 41–55.

[2] Baldacci, R., Bartolini, E., Mingozzi, A., 2011. An exact algorithm for the pickup and delivery problem with time windows. Operations Research 59(2), 414–426.

[3] Baldacci, R., Bartolini, E., Mingozzi, A., Roberti, R., 2010. An exact solution framework for a broad class of vehicle routing problems. Computational Management Science 7, 229–268.

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

[5] Batista-Galván, M., Riera-Ledesma, J.and Salazar-González, J., 2013. The traveling purchaser problem, with multiple stacks and deliveries: A branch-and-cut approach. Computers & Operations Research 40(8), 2103–2115.

[6] Battara, M., Cordeau, J.-F., Iori, M., 2014. Pickup-and-delivery problems for goods transportation. In: Toth, P., Vigo, D. (Eds.), Vehicle Routing: Problems, Methods, and Applications - Second Edition. Society for Industrial and Applied Mathematics, Philadelphia, Ch. 6, pp. 161–191.

[7] Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I., Laporte, G., 2007. Static pickup and delivery problems: A classification scheme and survey. TOP 15, 1–31.

[8] Carrabs, F., Cerulli, R., Cordeau, J.-F., 2007. An additive branch-and-bound algorithm for the pickup and delivery traveling salesman problem with LIFO or FIFO loading. INFOR 45(4), 223–238.

[9] Carrabs, F., Cordeau, J.-F., Laporte, G., 2007. Variable neighborhood search for the pickup and delivery traveling salesman problem with LIFO loading. INFORMS Journal on Computing 19(4), 618–632.

[10] Cherkesly, M., Desaulniers, G., Laporte, G., 2014. Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and last-in-first-out loading. Transportation Science. http://dx.doi.org/10.1287/trsc.2014.0535

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

[12] Cordeau, J.-F., Iori, M., Laporte, G., Salazar González, J., 2010. A branch-and-cut algorithm for the pickup and delivery traveling salesman problem with LIFO loading. Networks 55(1), 46–59.

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

[14] Côté, J.-F., Gendreau, M., Potvin, J.-Y., 2012. Large neighborhood search for the pickup and delivery traveling salesman problem with multiple stacks. Networks 60(1), 19–30.

[15] Desaulniers, G., Desrosiers, J., Solomon, eds., M. M., 2005. Column Generation. Springer, New York.

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

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

[18] Iori, M., Riera-Ledesma, J., 01/2015. Exact algorithms for the double vehicle routing problem with multiple stacks. Tech. rep., DIIS-Universidad de La Laguna.

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

[20] Kohl, N., Desrosiers, J., Madsen, O., Solomon, M., Soumis, F., 1999. 2-path cuts for the vehicle routing problem with time windows. Transportation Science 33(1), 101–116.

[21] Lusby, R., Larsen, J., Ehrgott, M., Ryan, D., 2010. An exact method for the double tsp with multiple stacks. International Transactions in Operational Research 17(5), 637–652.

[22] Naddef, D., Rinaldi, G., 2002. Branch-and-cut algorithms for the capacitated VRP. In: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications. Society for Industrial and Applied Mathematics, Philadelphia, Ch. 3, pp. 53–84.

[23] Parragh, S., Doerner, K., Hartl, R., 2008. A survey on pickup and delivery problems, Part I: Transportation between customers and depot. Journal für Betriebswirtschaft 58, 21–51.

[24] Parragh, S., Doerner, K., Hartl, R., 2008. A survey on pickup and delivery problems, Part II: Transportation between pickup and delivery locations. Journal für Betriebswirtschaft 58, 81–117.

[25] Petersen, H., Archetti, C., Speranza, M., 2010. Exact solutions to the double travelling salesman problem with multiple stacks. Networks 56(4), 229–243.

[26] Petersen, H., Madsen, O., 2009. The double travelling salesman problem with multiple stacks–formulation and heuristic solution approaches. European Journal of Operational Research 198(1), 139–147.

[27] Ropke, S., Cordeau, J.-F., 2008. Branch-and-cut-and-price for the pickup and delivery problem with time windows. Tech. rep., CIRRELT–2008–33, HEC Montréal, Montréal.

[28] Ropke, S., Cordeau, J.-F., 2009. Branch-and-cut-and-price for the pickup and delivery problem with time windows. Transportation Science 43(3), 267–286.

[29] Ropke, S., Cordeau, J.-F., Laporte, G., 2007. Models and branch-and-cut algorithms for pickup and delivery problems with time windows. Networks 49(4), 258–272.

[30] Savelsbergh, M., 1985. Local search in routing problems with time windows. Annals of Operations research 4(1), 285–305.