Partial Dominance in Branch-Price-and-Cut for the Basic Multi-Compartment Vehicle-Routing Problem

Katrin Heßler*,[a], Stefan Irnich[a]

[a] *Chair of Logistics Management, Gutenberg School of Management and Economics, Johannes Gutenberg University Mainz, Jakob-Welder-Weg 9, D-55128 Mainz, Germany.*

Abstract

We consider the exact solution of the basic version of the multiple-compartment vehicle-routing problem, i.e., a problem consisting of clustering customers into groups, routing a vehicle for each group, and packing demands of the visited customer uniquely into one of the vehicle's compartments. Compartments have a fixed size, and there are no incompatibilities between the transported items or between items and compartments. The objective is to minimize the total distance of all vehicle routes such that all customers are visited. We study the shortest-path subproblem that arises when exactly solving the problem with a branch-price-and-cut algorithm. For this subproblem, we compare a standard dynamic-programming labeling approach with a new one that utilizes a partial dominance. While the algorithm with standard labeling already struggles with relatively small instances, the one with partial dominance can cope with much larger instances.

Key words: vehicle routing, packing, shortest-path problem with resource constraints, dynamic-programming labeling, partial dominance

1. Introduction

In this paper, we consider the exact solution of the basic version of the multiple-compartment vehicle-routing problem, which extends the capacitated *vehicle-routing problem* (VRP, Toth and Vigo, 2014) by the task to uniquely assign every customer's demand to a compartment of the vehicle that performs the visit. The focus is the comparison of two different modeling and associated labeling approaches for the shortest-path problems with resource constraints that have to be solved repeatedly within a *branch-price-and-cut* (BPC) algorithm. The standard labeling approach is straightforward and directly assigns the customer demand to a compartment, while the new labeling approach represents all possible demand assignments within a single label. The latter labeling approach results in a poor direct one-to-one dominance between labels, but it allows the application of a new partial dominance. We derive criteria for reducing undesirable properties of the straightforward labeling approach related to symmetric and redundant intermediate solutions. Both the standard labeling but even more the labeling approach with partial dominance can successfully exploit these criteria.

The *basic multiple-compartment vehicle-routing problem* (BMCVRP) that we study here assumes that the vehicle fleet is homogeneous and that each vehicle has at least two separate loading spaces (called compartments) of fixed size. It is not allowed to split a customer's demand and load only parts of the demand into different compartments (either because goods are bulky or for operational/handling reasons, see Ostermeier *et al.* (2018); Cherkesly and Gschwind (2020); Heßler *et al.* (2021). The literature on multi-compartment vehicle routing is extensive, but excellent surveys are available, e.g., by Coelho and Laporte (2015) and Ostermeier *et al.* (2021). We therefore briefly distinguish the BMCVRP from other popular variants with multiple compartments.

*Corresponding author.
Email addresses: khessler@uni-mainz.de (Katrin Heßler), irnich@uni-mainz.de (Stefan Irnich)

In many applications, it is necessary to consider different commodities (goods, products, waste) that are to be distributed to (or collected from) customers. These commodities require transportation in separate and/or dedicated compartments, and one commodity can be incompatible with another commodity. In supermarket supply chains, temperature-sensitive groceries (fresh, ambient, and frozen) have to be distributed. Accordingly, the compartments provide different cooling zones, and sometimes the overall loading space is configurable into compartments of flexible size (Heßler, 2021). Moreover, compatibility between products (which products can be shipped in the same compartment?) and fixed assignments of products to compartments are predominant.

The replenishment problem of gas stations (Cornillier *et al.*, 2008) is often considered a periodic or inventory-routing VRP, in which different types of gas (regular and high-octane gas, diesel) must be transported in separate compartments. Coelho and Laporte (2015) distinguish four cases depending on whether customers allow multiple visits per period and on whether vehicles are equipped with gas meters, so that split deliveries are possible (otherwise, a full delivery implies that the entire content of a compartment must fit into the customer's underground tank).

Further important fields of application are the collection of urban waste (Henke *et al.*, 2015; Kiilerich and Wøhlk, 2017), the collection of agricultural products and the transportation of livestock (Oppen and Løkketangen, 2008; Lahyani *et al.*, 2015), and maritime transportation of cement and fuel (Christiansen *et al.*, 2011; Agra *et al.*, 2013).

Two loading spaces naturally arise when vehicles are composed of a truck and a trailer. However, the focus of truck-and-trailer VRPs is different (Rothenbächer *et al.*, 2018; Parragh and Cordeau, 2017), because they stress the need and possibility to decouple the trailer from the truck before a visit to a customer, in particular to those customers, where a combined vehicle is not allowed, e.g., due to limited maneuvering possibilities. The truck must later come back to the trailer's parking position and couple the trailer again. Important applications arise in row-milk collection (Drexl, 2007), where goods are typically divisible.

In the pickup-and-delivery problem with time windows and multiple stacks (Cherkesly *et al.*, 2016), the customer demands must be assigned uniquely to compartments. Note that here demands are point-to-point transportation requests, and the stacking constraints enforce the LIFO loading policy, i.e., that goods loaded last into a compartment must be delivered first from that compartment.

To summarize, we classify the BMCVRP according to the taxonomy of Ostermeier *et al.* (2021): The BMCVRP is a VRP with only one type of product, several demands of different customers are allowed to share a compartment (SC), vehicles have compartments of fixed size (FixedS), and a flexible assignment of this product to compartments (FlexA) is allowed (these are the compartment-related attributes). Only a single visit to a customer is allowed (SV), and therefore the demand of a customer is not split (UD) (these are the order fulfillment-related attributes). By duplicating customers that then reside at the same physical location, the BMCVRP also covers the case of possible multiple visits (MV), giving rise to the discrete split delivery VRP a.k.a commodity-constrained split delivery VRP (see Gschwind *et al.*, 2019).

We comment on the technical term *partial dominance* now. For a general introduction to BPC algorithms for VRPs we refer to (Costa *et al.*, 2019) and for modeling and solving *shortest-path problems with resource constraints* (SPPRC) to (Irnich and Desaulniers, 2005). The effectiveness of labeling algorithms strongly depends on the possibility to eliminate labels with the help of dominance rules. Recall that a label L is a (convenient) representation of a partial path $P(L)$ that starts at a given origin vertex. *Partial dominance* describes those dominance algorithms that use (in general) several labels for the elimination of a label. More precisely, a label L is dominated by $s \geq 1$ other labels $L^1, L^2, \ldots, L^s$ if for every feasible extension Q of its partial path $P = P(L)$ into a feasible origin-destination path (P, Q), there exists a feasible extension Q' of at least one of the other partial paths, say that of label L^r (for $r \in \{1, 2, \ldots, s\}$), into a feasible origin-destination path $p' = (P(L^r), Q')$ with smaller or identical cost compared to (P, Q).

Partial dominance has been used in relaxations of the elementary SPPRC: For the elimination 2-cycles (Houck *et al.*, 1980; Kohl *et al.*, 1999), i.e., short cycles of the form (i, j, i), the standard one-to-one dominance applies only to labels that are extended from the same predecessor vertex. A label can be jointly dominated by two other labels that have mutually different predecessor vertices, i.e., they all result from extensions from different vertices. Later, Irnich and Villeneuve (2006) generalize this partial dominance to k-cycle

elimination, i.e., all cycles of length up to k are forbidden. Here, not more than six different predecessor sequences are needed in 3-cycle elimination and not more than $k!(k-1)!$ different predecessor sequences in general. A problem-specific type of partial dominance has been introduced for the ng-route relaxation of the SPPRC subproblem of the minimum latency problem (Bulhões *et al.*, 2018).

For two-arc fixing using reduced costs, Desaulniers *et al.* (2020) present a labeling algorithm that relies on partial dominance. Moreover, vehicle scheduling and routing problems with SPPRC subproblems that have a tradeoff between two resources can benefit from a partial dominance as shown for airplane flight scheduling (Ioachim *et al.*, 1998, 1999), the active-passive VRP (Tilk *et al.*, 2018), and the VRP with convex inconvenience cost functions (convex node costs) defined on time windows (He *et al.*, 2019). The seminal paper (Desaulniers *et al.*, 1998) describes when and how linear tradeoffs between resources result from resource constraints in extensive path-based formulations solved with column-generation methods. The linear tradeoff imposes linear node/vertex costs, for which Desaulniers and Villeneuve (2000) provide a general approach using an implicit form of partial dominance. Even if we did not find works suggesting partial dominance for some applications, it should also be applicable when solving SPPRC subproblems of the following problems: the VRP with soft time windows (Liberatore *et al.*, 2010), the time-window assignment VRP (Spliet and Gabor, 2015), the electric VRP with time windows and partial recharges (Desaulniers *et al.*, 2016b), the split-delivery VRP with time windows (Desaulniers, 2010), the dial-a-ride problem with ride-time constraints (Gschwind and Irnich, 2015), the inventory-routing problem (Desaulniers *et al.*, 2016a), and the truck-and-trailer VRP with quantity-dependent transfer times (Rothenbächer *et al.*, 2018). This list should not be considered complete.

Overall, the contribution of this paper is the development of a powerful exact optimization algorithm for the BMCVRP, which is the basic problem synthesizing VRP and one-dimensional bin packing with multiple bin sizes (Correia *et al.*, 2008). The methodological contribution is the design and comparison of two shortest-path labeling algorithms for the subproblems that have to be repeatedly solved within the branch-price-and-cut algorithm. Both labeling algorithms benefit from new techniques that reduce inherent symmetry which can produce redundant labels. The computational experiments will reveal that the first and more standard labeling is strongly inferior to the second one, which relies on a new type of partial dominance.

The remainder of the paper is organized as follows. In Section 2, we formally define the BMCVRP, provide a route-based formulation, and explain how a tight lower bound on the fleet size can be computed. Section 3 provides two results on sorted and unsorted vectors important for symmetry reduction. The standard labeling and labeling algorithm with partial dominance are presented in Section 4. Results and interpretations of the computational experiments are provided in Section 5, before conclusions are drawn in Section 6.

2. Problem Definition

The BMCVRP can be formally defined over the undirected complete graph $G = (V, E)$ with vertex set V and edge set E. Let $N = \{1, 2, \ldots, n\}$ denote the set of customers and 0 the depot such that $V = N \cup \{0\}$. The vehicle fleet is homogeneous and housed at the depot comprising F vehicles. Each vehicle has $m \geq 2$ compartments indexed by $k \in C = \{1, 2, \ldots, m\}$. The compartment sizes are $\mathbf{Q} = (Q_1, Q_2, \ldots, Q_m)$ so that the total capacity of a vehicles is $Q = \sum_{k \in C} Q_k = \mathbf{1}^\top \mathbf{Q}$, where $\mathbf{1}$ denotes the vector $(1, 1, \ldots, 1) \in \mathbb{R}^m$. Each customer $i \in N$ has a demand given by the positive integer d_i. The demand is non-splitable in the sense that it must be uniquely assigned to one compartment of the vehicle that serves customer i. Let c_{ij} denote the routing cost for each $ij \in E$.

2.1. Route-based Formulation

The branch-price-and-cut algorithms that we explain and use later are based on a route-based model. A route is a cycle r in G containing the depot 0. We denote by $V(r)$ the set of vertices and by $E(r)$ the set of edges of r. The cost of a route is $c^r = \sum_{ij \in E(r)} c_{ij}$. A route is feasible if the demand $(d_i)_{i \in V(r) \setminus \{0\}}$ can be feasibly assigned to the compartments of one vehicles, i.e., there exists a partition $V(r) \setminus \{0\} = V_1 \cup V_2 \cup \ldots \cup V_m$

with $\sum_{i \in V_k} d_i \leq Q_k$ for all $k \in C$. In the following, the associated utilization $\mathbf{w} = (\sum_{i \in V_k} d_i)_{k \in C} = (\sum_{i \in V_1} d_i, \sum_{i \in V_2} d_i, \ldots, \sum_{i \in V_m} d_i)$ of the compartments is called a *packing*, an assignment of demands to a single compartment is called a *pattern*. A packing is feasible if $\mathbf{w} \leq \mathbf{Q}$ holds. The task of the BMCVRP is to find a set of feasible routes, where each route has an associated feasible packing, so that each customer $i \in N$ is served exactly once.

Let Ω denote the set of all feasible routes. The route-based set-partitioning formulation of the BMCVRP uses binary route variables λ^r for all routes $r \in \Omega$ and a non-negative integer variable f to count the number of vehicles that are employed:

$$\min \quad \sum_{r \in \Omega} c^r \lambda^r \tag{1a}$$

$$\text{subject to} \quad \sum_{r \in \Omega \,:\, i \in V(r)} \lambda^r = 1 \qquad \forall i \in N \tag{1b}$$

$$\sum_{r \in \Omega} \lambda^r - f = 0 \tag{1c}$$

$$F_{LB} \leq f \leq F \text{ and integer} \tag{1d}$$

$$\lambda^r \in \{0, 1\} \qquad \forall r \in \Omega. \tag{1e}$$

The objective (1a) is the minimization of the routing costs. The visit of all customers is guaranteed by constraints (1b). The coupling between the route variables and the vehicle-counting variable f is established via (1c). A valid lower bound F_{LB} on the number of necessary vehicles can be quickly computed as $F_{LB} = \lceil \sum_{i \in N} d_i / Q \rceil$. The solution of the respective bin-packing problem may provide an even tighter bound. The domains of the variables are given by (1d) and (1e).

2.2. Computation of a Fleet-Size Lower Bound

We comment on the computation of a tight lower bound $F^\star_{LB}$ for the minimal size of the vehicle fleet now. For the case of identical compartments, i.e., $Q_1 = Q_2 = \cdots = Q_m = Q/m$, we can compute such an exact lower bound by solving a bin-packing or cutting-stock problem with bins of capacity Q/m and demands $(d_i)_{i \in N}$. The computed number z_{BP} of bins can then be divided by m and rounded up to the next integer, i.e.,

$$F^\star_{LB} = \left\lceil \frac{z_{BP}}{m} \right\rceil .$$

For the case of compartments of different size, the following approach yields the exact lower bound. Let $\bar{C} \subset C$ be an index set (not unique) of all different capacity values, i.e., $\{Q_k : k \in \bar{C}\} = \{Q_k : k \in C\}$ and $Q_k \neq Q_{k'}$ for all $k, k' \in \bar{C}$ with $k \neq k'$. Further, let h_k denote the frequency, i.e., the number of compartments of size Q_k for all $k \in \bar{C}$, i.e., $h_k = |\{k' \in C : Q_k = Q_{k'}\}|$. For example, $(Q_1, Q_2, \ldots, Q_6) = (50, 50, 30, 20, 20, 20)$ has frequencies 2, 1, and 3 with $k \in \bar{C} = \{1, 3, 4\}$. Likewise, let $\bar{N}$ be an index set (not unique) of all different demand values occurring with frequencies b_i for $i \in \bar{N}$.

The model that we suggest uses copies of the arc-flow formulation of the *cutting-stock problem* (CSP) (Valerió de Carvalho, 1998), more specifically one copy for each compartment size given by $k \in \bar{C}$. Recall that the arc-flow formulation of the CSP uses a digraph in which each feasible pattern is in one-to-one correspondence to a source-to-sink path in the digraph. Hence, a source-to-sink flow through the digraph represents a solution to the CSP if sufficient flow passes through each subset of arcs that corresponds to a specific demand value b_i for all $i \in \bar{N}$.

Let $D^k = (V^k, A^k)$ be the digraph for the index $k \in \bar{C}$ with source vertex o^k and sink vertex d^k. The subset of arcs corresponding to the demand d_i for $i \in \bar{N}$ is denoted by $A^k(i)$. There are non-negative integer flow variables $\mathbf{x}^k = (x^k_{hj})_{(h,j) \in A^k}$. The non-negative integer variables z^k for $k \in \bar{C}$ indicate the total flow

through the network D^k. Finally, the variable z provides the objective value. The model is:

$$F^\star_{LB} = \quad \min \quad z \tag{2a}$$

$$\text{subject to} \quad h^k z \geq z^k \qquad \forall k \in \bar{C} \tag{2b}$$

$$\sum_{h\,:\,(h,i)\in A} x^k_{hi} - \sum_{j\,:\,(i,j)\in A} x^k_{ij} = \begin{cases} +z^k, & \text{if } i = o^k \\ -z^k, & \text{if } i = d^k \\ 0, & \text{otherwise} \end{cases} \qquad \forall k \in \bar{C}, i \in \bar{N} \tag{2c}$$

$$\sum_{k\in\bar{C}} \sum_{(h,j)\in A^k(i)} x^k_{hj} \geq b_i \qquad \forall i \in \bar{N} \tag{2d}$$

$$x^k_{hj} \geq 0 \text{ and integer} \qquad \forall k \in \bar{C}, (h,j) \in A^k \tag{2e}$$

$$z, z^k \geq 0 \qquad \forall k \in \bar{C}. \tag{2f}$$

The objective (2a) minimizes the total number of packings covering the entire demand. Constraints (2b) couple the objective value with the total flow through each network D^k for $k \in \bar{C}$. Due to the inequality, empty compartments without any assigned demand are possible. Hence, no additional loss arcs must be added to the network to allow empty compartments. Flow conservation and the coupling with the z^k-variables is accomplished via constraints (2c). The demand covering constraints (2d) differ from those of the original CSP because they refer to all networks D^k for $k \in \bar{C}$ together. The domains of the variables are stated in (2e) and (2f).

Example 1. *Consider an instance of the BMCVRP in which vehicles have $m = 2$ compartments of size $\mathbf{Q} = (11, 7)$ so that the total vehicle capacity is $Q = 18$. The customers $N = \{1, 2, \ldots, 14\}$ have demands $d_1 = \cdots = d_4 = 8$ and $d_5 = \cdots = d_{14} = 4$. The total demand is $4 \cdot 8 + 10 \cdot 4 = 72$.*

The trivial lower bound of the fleet size is $\lceil 72/18 \rceil = 4$. The bin-packing bound, which is the exact lower bound on the number of vehicles for the capacitated VRP, is 5, because every feasible packing into bins of size $Q = 18$ gives a loss of at least 2 units and any greedy packing produces a solution with 5 bins. Finally, model (2) provides the exact lower bound $F^\star_{LB} = 6$. Note that four compartments of size $Q_1 = 11$ are exclusively occupied with the first four demands $d_1 = \cdots = d_4 = 8$. The other ten demands $d_5 = \cdots = d_{14} = 4$ either occupy a second compartment of size $Q_2 = 7$ or two of them can be packed together into the first compartment of size $Q_1 = 11$. A possible solution therefore consists of four packings $(8, 4)$ and two packings $(4 + 4, 4)$. □

Note that also refined pseudo-polynomial formulations for the CSP and bin-packing problem can be used (Delorme and Iori, 2020, provide an overview). Since the computation of the lower bound is not time-critical in our application, we retain the simple model and hereby keep the description clear.

3. Some Properties of Sorted and Unsorted Vectors

In this short section, we collect some results on sorted and unsorted vectors that will be helpful to reduce symmetry in the label extension step, the dominance algorithm, and the label merge step in bidirectional labeling presented in the next section.

We start with defining some notation. The set of all possible permutations of $\{1, 2, \ldots, m\}$ is denoted by Σ_m. For a vector $\mathbf{a} = (a_1, a_2, \ldots, a_m) \in \mathbb{R}^m$ and a permutation $\pi \in \Sigma_m$, we denote by $\mathbf{a}_\pi$ the vector $(a_{\pi(1)}, a_{\pi(2)}, \ldots, a_{\pi(m)})$. Two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ are *permutation-equivalent* if there exists a permutation $\pi \in \Sigma_m$ such that $\mathbf{a}_\pi = \mathbf{b}$. A vector $\mathbf{a} = (a_1, a_2, \ldots, a_m) \in \mathbb{R}^m$ is (componentwise non-decreasingly) *sorted* if $a_1 \leq a_2 \leq \cdots \leq a_m$. For a vector $\mathbf{a} \in \mathbb{R}^m$, the (unique) permutation-equivalent and sorted vector is denoted by $\mathbf{a}_\leq \in \mathbb{R}^m$. For two permutations $\pi, \tau \in \Sigma_m$, the concatenation $\tau \circ \pi \in \Sigma_m$ gives

$$\mathbf{a}_{\tau\circ\pi} = (a_{\tau\circ\pi(1)}, a_{\tau\circ\pi(2)}, \ldots, a_{\tau\circ\pi(m)}) = (a_{\tau(\pi(1))}, a_{\tau(\pi(2))}, \ldots, a_{\tau(\pi(m))}) = (\mathbf{a}_\pi)_\tau.$$

Proposition 1. *Let $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ be two vectors. The following two statements are equivalent:*

(1) there exists a permutation $\pi \in \Sigma_m$ *such that* $\mathbf{a}_\pi \leq \mathbf{b}$*;*
(2) $\mathbf{a}_{\leq} \leq \mathbf{b}_{\leq}$.

Proof. "(1) $\Rightarrow$ (2)": Let $\mathbf{a}_\pi \leq \mathbf{b}$ for the permutation $\pi \in \Sigma_m$. We first sort the inequalities of the componentwise comparison $\mathbf{a}_\pi \leq \mathbf{b}$ such that the values $a_1, \ldots, a_m$ on the left-hand side are non-decreasing. For this purpose, let $\tau \in \Sigma_m$ be the permutation that produces the sorted vector, i.e., $\mathbf{a}_{\tau\circ\pi} = \mathbf{a}_{\leq}$. The sorting guarantees

$$a_{\tau\circ\pi(i)} \leq a_{\tau\circ\pi(j)} \qquad \text{for all } i, j \in \{1, 2, \ldots, m\} \text{ with } i < j, \tag{3}$$

Moreover $\mathbf{a}_\pi \leq \mathbf{b}$ implies

$$\mathbf{a}_{\leq} = \mathbf{a}_{\tau\circ\pi} \leq \mathbf{b}_\tau. \tag{4}$$

If $\mathbf{b}_\tau$ is already sorted, i.e., $\mathbf{b}_\tau = \mathbf{b}_{\leq}$, then $\mathbf{a}_{\leq} \leq \mathbf{b}_{\leq}$ follows immediately from (4). Otherwise, $\mathbf{b}_\tau$ is *not* sorted, i.e., there exist two indices $i, j \in \{1, 2, \ldots, m\}$ such that $i < j$ and $b_{\tau(i)} > b_{\tau(j)}$. Then,

$$a_{\tau\circ\pi(i)} \overset{(3)}{\leq} a_{\tau\circ\pi(j)} \overset{(4)}{\leq} b_{\tau(j)} \qquad \text{and} \qquad a_{\tau\circ\pi(j)} \overset{(4)}{\leq} b_{\tau(j)} < b_{\tau(i)}.$$

Both inequalities mean that $b_{\tau(i)}$ and $b_{\tau(j)}$ (two values on the right-hand side of (4)) can be swapped without sacrificing $\mathbf{a}_{\leq} \leq \mathbf{b}_\tau$. (Formally, τ is replaced by $\tau \circ (i, j)$, where $(i, j) \in \Sigma_m$ denotes the permutation that swaps i and j.) The repeated application of such swaps for pairs $i < j$ and $b_{\tau(i)} > b_{\tau(j)}$ leads to a sorted vector $\mathbf{b}_\tau$ establishing the result $\mathbf{a}_{\leq} \leq \mathbf{b}_\tau = \mathbf{b}_{\leq}$.

"(2) $\Leftarrow$ (1)": Let $\mathbf{a}_{\leq} \leq \mathbf{b}_{\leq}$. Then there exist two permutations $\rho, \tau \in \Sigma_m$ such that $\mathbf{a}_{\leq} = \mathbf{a}_\rho$ and $\mathbf{b}_{\leq} = \mathbf{b}_\tau$. It follows $\mathbf{a}_\rho = \mathbf{a}_{\leq} \leq \mathbf{b}_{\leq} = \mathbf{b}_\tau$, which implies $\mathbf{a}_{\tau^{-1}\circ\rho} \leq \mathbf{b}$. Hence, $\pi = \tau^{-1} \circ \rho \in \Sigma_m$ is the permutation that establishes the result. □

A vector $\mathbf{b} = (b_1, b_2, \ldots, b_m) \in \mathbb{R}^m$ is (componentwise) *non-increasingly sorted* if $b_1 \geq b_2 \geq \cdots \geq b_m$. For a vector $\mathbf{b} \in \mathbb{R}^m$, the (unique) permutation-equivalent and non-increasingly sorted vector is denoted by $\mathbf{b}_{\geq} \in \mathbb{R}^m$.

Proposition 2. *Let* $\mathbf{a}, \mathbf{b} \in \mathbb{R}^m$ *be two vectors, and let* $Q \in \mathbb{R}$ *be a real value. The following two statements are equivalent:*
(1) there exists a permutation $\pi \in \Sigma_m$ *such that* $\mathbf{a}_\pi + \mathbf{b} \leq Q\mathbf{1}$*;*
(2) $\mathbf{a}_{\leq} + \mathbf{b}_{\geq} \leq Q\mathbf{1}$.

Proof. We can re-write $\mathbf{a}_\pi + \mathbf{b} \leq Q\mathbf{1}$ into $\mathbf{a}_\pi \leq Q\mathbf{1} - \mathbf{b}$, and likewise, $\mathbf{a}_{\leq} + \mathbf{b}_{\geq} \leq Q\mathbf{1}$ into $\mathbf{a}_{\leq} \leq Q\mathbf{1} - \mathbf{b}_{\geq}$. Since the latter right-hand side $Q\mathbf{1} - \mathbf{b}_{\geq}$ is equal to $(Q\mathbf{1} - \mathbf{b})_{\leq}$, the equivalence is a direct consequence of Proposition 1. □

Remark 1. *When replacing* $Q\mathbf{1} = (Q, \ldots, Q)$ *by* $\mathbf{Q} = (Q_1, Q_2, \ldots, Q_m)$ *with* $Q_i \neq Q_j$ *for at least one pair* $i \neq j$*, statements (1) and (2) in Proposition 2 are no longer equivalent. As a counterexample, consider* $\mathbf{a} = (2, 3)$*,* $\mathbf{b} = (1, 3)$*, and* $\mathbf{Q} = (3, 6)$*. Then* $\mathbf{a}_\pi + \mathbf{b} \leq \mathbf{Q}$ *for identity permutation* π *but* $\mathbf{a}_{\leq} + \mathbf{b}_{\geq} = (5, 4) \not\leq \mathbf{Q}$.

4. Solution of the Subproblem

According to Costa *et al.* (2019), 'the leading exact algorithms for solving many classes of VRPs are branch-price-and-cut algorithms'. They summarize: 'A branch-price-and-cut algorithm is a branch-and-bound algorithm where the lower bounds are computed by column generation and the cutting planes are added to strengthen the linear relaxations encountered in the search tree'. For a general introduction to column generation and BPC algorithms, we refer to (Desaulniers *et al.*, 1998, 2005; Lübbecke and Desrosiers, 2005).

For presenting the labeling algorithm, it is advantageous to work on a directed graph, because label extensions are directed anyway and splitting the depot into an origin and a destination vertex is necessary. We therefore denote by $(N \cup \{0, 0'\}, A)$ the associated complete directed graph with origin vertex 0, destination

vertex $0'$, vertex set $N \cup \{0, 0'\}$, and arc set A. The routing costs in the BMCVRP are symmetric implying that $c_{ij} = c_{ji}$ for all arcs $(i, j) \in A \cap (N \times N)$ between two customers and $c_{i0'} = c_{0i}$ for all arcs connecting a customer $i \in N$ with the depot.

For this work, it suffices to know that the linear relaxation of a restricted route-based model (1), in which only a subset $\bar{\Omega} \subset \Omega$ of all routes are initially considered (the *restricted master program*, RMP) and binary route variables are replaced by $\lambda^r \geq 0$ for all $r \in \Omega$, is solved by column generation and strengthened with the help of *capacity cuts* (CCs) and *subset-row inequalities* (SRIs, Jepsen *et al.*, 2008). We use the *ng*-route relaxation (Baldacci *et al.*, 2011) of the elementary SPPRC subproblem with a neighborhood $N_i \subset N$ for each vertex $i \in N \cup \{0, 0'\}$. Since the *ng*-route relaxation allows cycles in routes, we need to extend the notation. Let the integer coefficient a_i^r denote the number of times that route $r \in \Omega$ visits vertex $i \in V$ and h_{ij}^r denote the number of times that route $r \in \Omega$ traverses arc $(i, j) \in A$.

Capacity cuts require a minimum flow of vehicles trough the arcs $\delta^+(C) = \{(i, j) \in A : i \in C, j \notin C\}$ for a subset $C \subset N$ of the customers. As for the overall fleet lower bound, we have the choice between different lower bounds, with a tradeoff between the computational effort to compute the bound and its strength. We use the simple-to-compute lower bound $F_{LB}(C) = \lceil \sum_{i \in C} d_i / Q \rceil$ for the following capacity cuts:

$$\sum_{r \in \bar{\Omega}} \sum_{\substack{(i,j) \in \\ A(r) \cap \delta^+(C)}} h_{ij}^r \lambda^r \geq F_{LB}(C) \qquad \forall C \subseteq N, |C| \geq 2$$

Tight bounds—in general stronger than $F_{LB}(C)$—result from the solution of a respective bin-packing problem or problem (2) using the subset C instead of the customer set N (see Section 2.2).

For SRIs, we use row subsets $S \subset N$ of cardinality three only. Let $\mathscr{S}$ denote the collection of subsets $S \subset N$, i.e., $S \in \mathscr{S}$ is used to refer to the SRIs present in the RMP:

$$\sum_{r \in \bar{\Omega}} \left\lfloor \frac{\sum_{i \in S} a_i^r}{2} \right\rfloor \lambda^r \leq \left\lfloor \frac{|S|}{2} \right\rfloor \qquad \forall S \in \mathscr{S} \tag{5}$$

The column-generation subproblem (a.k.a. pricing problem) must identify a negative reduced cost route, if one exists. To this end, let $(\nu_i)_{i \in N}$ be the dual prices of the partitioning constraints (1b), let $\nu_0 = \nu_{0'}$ be the dual price of the coupling constraint (1c), and let $(\sigma_S)_{S \in \mathscr{S}}$ be the dual prices of the SRIs (5).

The reduced cost of a route $r \in \Omega$ is then

$$\tilde{c}^r = c^r - \nu_0 - \sum_{i \in N} a_i^r \nu_i - \sum_{S \in \mathscr{S}} \left\lfloor \frac{\sum_{i \in S} a_i^r}{2} \right\rfloor \sigma_S = \sum_{(i,j) \in A} h_{ij}^r \tilde{c}_{ij} - \sum_{S \in \mathscr{S}} \left\lfloor \frac{\sum_{i \in S} a_i^r}{2} \right\rfloor \sigma_S$$

with reduced cost $\tilde{c}_{ij} = c_{ij} - (\nu_i + \nu_j)/2$ for all arcs $(i, j) \in A$. Note that also the reduced costs are symmetric, i.e., $\tilde{c}_{ij} = \tilde{c}_{ji}$.

Labels L of the two different labeling algorithms that we present share the following attributes: the reduced cost $\tilde{c}(L)$ accumulated along the partial path $P = P(L)$, the total demand $w(L)$ accumulated along P, the binary attribute $sri_S(L)$ of a SRI for $S \in \mathscr{S}$, and the *ng*-neighborhood-related attribute $ng_v(L)$ for every $v \in N$. At the origin vertex 0, all attributes are set to zero, i.e., the initial label is $L_0 = (0, 0, \mathbf{0}, \mathbf{0})$. When extending a label L over an arc $(i, j) \in A$, the new label L' of the partial path $P' = (P, j) = P(L')$ has the following components:

$$\tilde{c}(L') = \tilde{c}(L) + \tilde{c}_{ij} - \sum_{\substack{S \in \mathscr{S} : j \in S, \\ sri_S(L) = 1}} \sigma_S$$

$$w(L') = w(L) + d_j$$

$$ng_v(L') = \begin{cases} ng_v(L) + 1, & \text{if } j = v \\ ng_v(L), & \text{if } v \in N_j \setminus \{j\} \\ 0, & \text{otherwise} \end{cases} \qquad \forall v \in N$$

$$sri_S(L') = \begin{cases} 1 - sri_S(L), & \text{if } j \in S \\ sri_S(L), & \text{otherwise} \end{cases} \qquad \forall S \in \mathscr{S}$$

The partial path $P(L')$ is feasible, if

$$w(L') \leq Q \qquad \text{and} \qquad ng_v(L) \leq 1 \quad \forall v \in N, \\ \text{and the associated packing(s) is (are) feasible.} \tag{6}$$

Necessary conditions for a label L^1 dominating a label L^2, both resident at the same vertex, are

$$\tilde{c}(L^1) - \sum_{\substack{S \in \mathscr{S}: \\ sri_S(L^1)=1, \\ sri_S(L^2)=0}} \sigma_S \leq \tilde{c}(L^2) \qquad \text{and} \qquad w(L^1) \leq w(L^2) \qquad \text{and} \qquad ng_v(L^1) \leq ng_v(L^2), \quad \forall v \in N. \tag{7}$$

We apply a version of bidirectional labeling (Righini and Salani, 2008; Tilk *et al.*, 2017) that exploits the symmetry of the pricing problem. Note that the dual prices of all robust constraints including those resulting from branching, see Section 5.2, are distributed in a symmetric manner onto the arcs of the digraph defining the pricing problem. As a result, the bidirectional labeling requires the computation of partial paths only for one direction. Labeling in the opposite direction would yield identical paths just ending at $0'$ and with reverse arc orientation. Hence, we compute only forward labels L using the total load onboard the vehicle as the critical resource, i.e., the attribute $w(L) \in \mathbb{R}$. Label extension ends, if either the destination vertex $0'$ is reached or the total load $w(L)$ exceeds the half-way point $H = Q/2$. This implicit bidirectional techniques has already been successfully implemented and applied, e.g., in (Bode and Irnich, 2012; Goeke *et al.*, 2019; Gschwind *et al.*, 2019).

In the sequel, we focus on the question of how to represent, manipulate, and check the feasibility of packings, in particular, condition (6), and how to extend the dominance (7) so that it considers packings consistently. We compare a standard labeling algorithm with a new labeling approach using partial dominance. They differ in the following fundamental underlying modeling assumptions:

- In the standard labeling algorithm, a label uniquely determines a partial path and also a packing, i.e., the information which demands of customers visited along the path are assigned to which compartment. Hence, a label represents a combination of a partial path and an associated packing. The modeling assumption also implies that the label extension step, where a label resident at a vertex $i \in V$ is extended over outgoing arcs $(i, j) \in A$, must generally create *multiple* labels for every vertex j. They represent the same partial path but different packings. Indeed, up to m new feasible labels can result depending on whether the demand of customer j fits only into some or all compartments. We discuss how symmetry reduction can substantially empower a naive implementation of standard labeling in Section 4.1.
- In the new labeling approach, a label represents a partial path for which (at least) one feasible packing exists. The decision about the concrete packing finally used is, however, postponed until the partial path reaches the destination. In this way, one label can represent many alternative packings. However, being more general comes at the cost of a more intricate dominance between labels. For example, it can happen now that some packings of one label are dominated by some packings of a second label, but some other packings are not dominated. In such a situation, the first label cannot be discarded directly. Several dominating labels together may however justify the elimination of the label. Partial dominance is the technique that can cope adequately with these complications, as we explain in Section 4.2.

4.1. Standard Labeling

Recall that the total vehicle capacity Q is divided into m compartments of size $\mathbf{Q} = (Q_1, Q_2, \ldots, Q_m)$ with $Q = \sum_{k=1}^{m} Q_k$. In the standard labeling approach, we explicitly store the load onboard each compartment, the packing, in a vector $\mathbf{w}(L) = (w_1, w_2, \ldots, w_m) \in \mathbb{R}^m$ for each label L. The label L_0 for the initial partial path $P(L_0) = (0)$ has a packing given by the null vector $\mathbf{w}(L_0) = \mathbf{0} \in \mathbb{R}^m$.

When extending a label L resident at vertex $i \in V$ along an arc $(i, j) \in A$, we create up to m new labels L', one for each compartment $k \in C$. For $k \in C$, the new label L' has the packing $\mathbf{w}(L') = (w_1, w_2, \ldots, w_k + d_j, \ldots, w_m) = \mathbf{w}(L) + d_j \mathbf{u}^k$, where $\mathbf{u}^k \in \mathbb{R}^m$ is the kth unit vector. The new packing is feasible if $\mathbf{w}(L') \leq \mathbf{Q}$.

Example 2. *We consider an instance with three customers 1,2, and 3 with demand* $d_1 = 2$, $d_1 = 3$, *and* $d_3 = 2$ *and vehicles with* $m = 2$ *different compartments of capacity* $\mathbf{Q} = (5, 10)$. *Along the path* $(0, 1, 2, 3)$, *a naive label extension procedure creates, out of the initial label* L_0, *new labels with the following packings:*

- *two labels for customer 1 with packings* $(0, 2)$ *and* $(2, 0)$*;*
- *four labels for customer 2 with packings* $(0, 5)$, $(2, 3)$, $(3, 2)$, *and* $(5, 0)$*;*
- *seven labels for customer 3 with packings* $(0, 7)$, $(2, 5)$ *(2 times),* $(3, 4)$, $(4, 3)$, *and* $(5, 2)$ *(2 times).*

Note that for customer 3 the packing $(7, 0)$ *is infeasible for the compartment sizes* $\mathbf{Q} = (5, 10)$. □

The example underlines the undesirable effect of identical and symmetric packings. While identical packings are clearly redundant, symmetric packings are not obviously redundant. In Example 2, the packing $(0, 7)$ for the path $(0, 1, 2, 3)$ can only be constructed from $(0, 5)$ and not from $(5, 0)$.

The consideration of sorted packings and sorted capacity vectors $\mathbf{Q} = \mathbf{Q}_{\leq}$ substantially reduces the occurrence of identical and symmetric packings. More precisely, we only allow labels L with sorted packings $\mathbf{w}(L) = \mathbf{w}_{\leq}(L)$. After an extension along an arc $(i, j) \in A$ assigning the demand d_j to compartment $k \in C$, the resulting packing $\mathbf{w}(L) + d_j\mathbf{u}^k$ may be unsorted. We distinguish the two cases of identical and different compartment size. For identical compartments, the feasibility test $\mathbf{w}(L)_k + d_j \leq Q/m$ can be performed adhoc. For compartments of different size, $\mathbf{w}(L) + d_j\mathbf{u}^k$ is first sorted creating the new packing $\mathbf{w}(L') = (\mathbf{w}(L) + d_j\mathbf{u}^k)_{\leq}$. In a second step, the feasibility test $\mathbf{w}(L') \leq \mathbf{Q}$ is performed. The correctness of this procedure (guaranteeing that no possible packings are missing) is ensured by Proposition 1.

When relying on sorted packings, only one extension needs to be performed per group of identical entries in the packing vector $\mathbf{w}(L)$. Otherwise identical packings would be created with the sorting.

Example 3. *(cont'd from Example 2) The initial packing* $(0, 0)$ *has two identical entries. Therefore, only one packing* $(0, 2) = (2, 0)_{\leq}$ *is created for customer 1. Overall, the restriction to sorted packings results in:*

- *one label for customer 1 with packing* $(0, 2)$*;*
- *two labels for customer 2 with packings* $(0, 5)$ *and* $(2, 3)$*;*
- *three labels for customer 3 with packings* $(0, 7)$, $(2, 5)$ *(2 times), and* $(3, 4)$.

Note that the two labels with identical packings $(2, 5)$ *cannot be avoided in standard labeling, because they result from the two different and independent labels at the preceeding customer 2, i.e., those with the packings* $(0, 5)$ *and* $(2, 3)$. □

One-to-one dominance between labels is straightforward due to Proposition 1:

Dominance 1. *For two labels* L^1 *and* L^2 *representing partial paths ending at the same vertex,* L^1 *dominates* L^2 *if* $\mathbf{w}(L^1) \leq \mathbf{w}(L^2)$ *and* (7) *(dominance on other resources).*

To identify negative reduced cost routes, we then consider the labels L that reach the destination vertex $0'$ resulting at most half-fully loaded vehicle routes. Moreover, we consider all pairs (L, L') of (forward) labels residing at the same vertex j. It is tested whether the concatenation of partial path $P = P(L)$ and the revered partial path $P' = P(L')$ produce a feasible route $r = (P, \text{reverse}(P'))$. A precondition to avoid identical results is the requirement $w(L) \geq H$. Since both labels L and L' already incorporate the demand of vertex j, it is now convenient to also consider the predecessor label $L'' = pred(L')$ of label L'. This idea of considering the predecessor L'' of the backward label L' has been used in (Tilk *et al.*, 2018) and several subsequent works. The negativity of the reduced cost and the feasibility of the resulting path r regarding the total load and the ng-route relaxation are to be tested with

$$\tilde{c}(L) + \tilde{c}(L') + \sum_{\substack{S \in \mathscr{S} : j \in S \\ sri_S(L)=0, \\ sri_S(L')=0}} \sigma_S < 0 \qquad \text{and} \qquad w(L) + w(L'') \leq Q$$

$$\text{and} \qquad ng_v(L) + ng_v(L') \leq 1, \quad \forall v \in N \setminus \{j\}. \tag{8}$$

Conditions (8) are necessary but not sufficient because packings are not tested yet. Regarding the combination of the two packings $\mathbf{w}(L)$ and $\mathbf{w}(L'')$, we distinguish the cases of identical and different compartments:

- For identical compartments, the necessary and sufficient condition is $\mathbf{w}(L) + \mathbf{w}(L'')_{\geq} \leq Q/m\,\mathbf{1}$, which results from Proposition 2.
- For different compartment size, we are not aware of a straightforward test. We need to find (if existent) two one-to-one mappings between compartments, i.e., permutations $\pi, \tau \in \Sigma_m$ such that $w_{\pi(k)}(L)+w_{\tau(k)}(L'') \leq Q_k$ holds for all $k \in C$. By Proposition 1, it suffices to test $(\mathbf{w}_{\pi}(L)+\mathbf{w}(L''))_{\leq} \leq \mathbf{Q}$ for all $\pi \in \Sigma_m$ exploiting that $\mathbf{Q} = \mathbf{Q}_{\leq}$ is sorted.

All components of the standard labeling approach, i.e., label extension, feasibility test, dominance, and merge procedure are defined now.

4.2. Labeling with Partial Dominance

The labeling algorithm that we describe now relies on the modeling approach in which a label represents a partial path. All feasible packings for this partial path are considered together. For this purpose, we define $\mathscr{W}(L)$ as the set of all feasible packings possible for the partial path $P(L)$ represented by the label L. We show that partial dominance can be established between labels by manipulating their sets $\mathscr{W}(L)$.

We assume that all packings $\mathbf{w} \in \mathscr{W}(L)$ are sorted, i.e., $\mathbf{w} = \mathbf{w}_{\leq}$, and also that sets $\mathscr{W}(L)$ of packings are sorted lexicographically.

The label L_0 for the initial partial path $P(L_0) = (0)$ has the set $\mathscr{W}(L_0) = \{\mathbf{0}\}$ of packings. When extending a label L over an arc $(i, j) \in A$, a single new label L' is generated (if feasible) as follows. For each $\mathbf{w} \in \mathscr{W}(L)$, the up to m possible resulting packings are created as in the standard labeling, feasibility is tested, and feasible packings are inserted into a tentative collection $\mathscr{W}(L')$. In this process, identical packings are immediately eliminated. Hence, we implement $\mathscr{W}(L)$ as a data structure which allows to (1) sort the packings in a lexicographically non-decreasing fashion, (2) eliminate identical packings, and (3) eliminate a selected subset of packings. Note that the identification of identical packings is trivial after a lexicographical sorting.

Example 4. *For the extension of a label L over the arc $(i, j) \in A$, let the set of packings be $\mathscr{W}(L) = \{(0, 2), (1, 1)\}$ and the demand at the head vertex be $d_j = 1$. The resulting new label is denoted by L'. The intermediate results is $\mathscr{W}(L') = \{(1, 2), (0, 3), (1, 2)\}$, while after sorting and elimination of duplicate packings the set becomes $\mathscr{W}(L') = \{(0, 3), (1, 2)\}$.*

The main difference between standard labeling and labeling with partial pricing is the dominance:

Dominance 2. *For two labels L^1 and L^2 representing partial paths ending at the same vertex, L^1 partially dominates L^2 if there is at least one pair of packings $\mathbf{w} \in \mathscr{W}(L^1)$ and $\mathbf{w}' \in \mathscr{W}(L^2)$ with $\mathbf{w} \leq \mathbf{w}'$ and (7) (dominance on other resources). In this case, all such packings $\mathbf{w}'$ can be eliminated from $\mathscr{W}(L^2)$. If the set $\mathscr{W}(L^2)$ becomes empty, the label L^2 is completely dominated and can be discarded.*

The representation of the packings in the set $\mathscr{W}(L)$ is redundant. The point is that all packings $\mathbf{w} \in \mathscr{W}(L)$ have the same total load $\mathbf{1}^{\top}\mathbf{w} = w(L)$. One can therefore use the attribute $w(L)$ and store only $m - 1$ components of the packings $\mathbf{w} \in \mathscr{W}(L)$. We use the *set of truncated packings* storing only the first $m - 1$ components.

4.2.1. The Case of Two Compartments

In particular for the case of only $m = 2$ compartments, we can exploit sets of truncated packings, which then contains numbers w_1 only. We can reconstruct the second component $w_2 = w(L) - w_1$ so that the packing is $\mathbf{w} = (w_1, w(L) - w_1)$. We next explain how label extension, dominance test, and merge procedure become simpler with the truncated sets of packings.

Label Extension. Algorithm 1 describes the extension of a label L over the arc $(i, j) \in A$. In Step 1, the new total load $w = w(L')$ is computed for the new partial path ending at vertex j. The idea is now to merge two sets of packings in which the new demand d_j is either added to the second compartment so that the value w_1 is just copied for every $w_1 \in \mathscr{W}(L)$, or added to the first compartment resulting in the value $w_1 + d_j$. The direct use of the values $w_1 + d_j$ for all $w_1 \in \mathscr{W}(L)$ would however be incorrect, because the packing could

Algorithm 1: Label Extension for $m = 2$

Input : label L and arc (i, j)
1 $w := w(L) + d_j$
2 $S1 := \{w_1 : w_1 \in \mathscr{W}(L), w - w_1 \leq Q_2\}$
3 $S2 := \{w_1 + d_j : w_1 \in \mathscr{W}(L), w_1 + d_j \leq w/2, w_1 + d_j \leq Q_1\}$
4 $S3 := \{w - w_1 - d_j : w_1 \in \mathscr{W}(L), w_1 + d_j > w/2, w - w_1 - d_j \leq Q_1\}$
5 Revert($S3$)
6 $\mathscr{W}' :=$ MergeWithoutDuplicates($S1, S2, S3$)
Output: $w(L') := w$ and $\mathscr{W}(L') := \mathscr{W}'$

Algorithm 2: Partial Dominance for $m=2$

Input : labels L^1 and L^2
1 $w_1 := \mathscr{W}(L^1)$.first()
2 $w'_1 := \mathscr{W}(L^2)$.first()
3 **while** $w_1 \neq \mathscr{W}(L^1).end()$ *and* $w'_1 \neq \mathscr{W}(L^2).end()$ **do**
4 next $:= \mathscr{W}(L^2)$.next(w'_1)
5 **while** $w_1 \leq w'_1$ **do**
6 **if** $w(L^1) - w_1 \leq w(L^2) - w'_1$ **then**
7 Remove w'_1 from $\mathscr{W}(L^2)$;
8 Break
9 **if** $\mathscr{W}(L^1).next(w_1) > w'_1$ **then**
10 Break
11 $w_1 := \mathscr{W}(L^1)$.next(w_1)
12 $w'_1 :=$ next
Output: (possibly) modified $\mathscr{W}(L^2)$

be unsorted, i.e., $w_1 + d_j > w_2 = w(L') - (w_1 + d_j)$. To distinguish between a packing for which entries must be retained or flipped, the values $w_1 + d_j$ and $w(L')/2$ must be compared. In summary, three sorted lists $S1$, $S2$, and $S3$ of values are computed in Steps 2–4, where the last list is sorted decreasingly. The reversal of $S3$ (Step 5) afterwards allows to merge three increasingly sorted lists of values in Step 6, where the insertion of duplicate entries can be suppressed. The result is a linear-time label-extension procedure with complexity $\mathcal{O}(|\mathscr{W}(L)|)$.

Partial Dominance Procedure. The second algorithmic component that can exploit the implicit representation with the set of truncated packings is the dominance test. Dominance still relies on the simple packing-wise comparison using Dominance 2. Algorithm 2 shows how the set of truncated packings of label L^2 has to be modified due to a possible partial dominance between labels L^1 and L^2. We assume that the sets $\mathscr{W}(L)$ of truncated packings are sorted and can be traversed (using `first()` to refer to the first (smallest) element, `next(`w_1`)` to refer to the element behind w_1, and `end()` to check whether the end of the truncated set of packings has been reached). The outer loop (Steps 3–12) traverses the values w'_1 of the set of truncated packings of L_2, while the inner loop (Steps 5–11) traverses values w_1 of the set of truncated packings of L_1 maintaining the invariant $w_1 \leq w'_1$. Hence, the overall run-time complexity is linearly bounded by $\mathcal{O}(|\mathscr{W}(L^1)| + |\mathscr{W}(L^2)|)$. We explain in Section 4.2.2 how an alternative implementation with a binary search can further improve the run time in the average case.

Bidirectional Labeling and Merge Procedure. The merge procedure can also exploit the two-compartment case with truncated sets of packings. Recall that for the merge we consider a forward label L and the predecessor $L'' = pred(L')$ of another backward (=forward) label L' in order to not count the demand of customer j twice. We assume that the necessary conditions (8) for a feasible merge have already been tested and are fulfilled. As another fast necessary condition, we check that the slack $\Delta = Q - w(L) - w(L'')$ in the total-capacity constraint is non-negative. For $\Delta \geq 0$, we distinguish two cases:

For identical compartments, i.e., $Q_1 = Q_2 = Q/2$, any combination of $w_1 \in \mathscr{W}(L)$ and $w''_1 \in \mathscr{W}(L'')$ is feasible if and only if $Q/2 - w_1 - w''_2 \leq \Delta$ according to Proposition 2 (due to $\Delta \geq 0$ both inequalities $w_1 + w''_2 \leq Q/2$ and $w_2 + w''_1 \leq Q/2$ hold true then). This merge condition means that w_1 and w''_2 must fit together into a compartment and that the unused capacity $Q/2 - w_1 - w''_2$ should be kept as small as possible. The latter observation is used in Algorithm 3, where w''_2 is computed as $w(L'') - w''_1$ in Step 8. As a result, the main loop in Steps 7–14 either increases w_1 or decreases w''_2 (by increasing w''_1).

For compartments of different size, we can no longer exploit Proposition 2. Four subcases have to be tested as shown in the first column of Table 1. The four cases can be checked in a similar way as described in Algorithm 3 (the first case is identical to Algorithm 3 with $Q/2$ replaced by Q_1 in Step 8). The necessary modifications for the three other cases are summarized in the second and third column of Table 1. On the

Algorithm 3: Merge Procedure for Identical Compartments and $m = 2$

Input : labels L and L' at same vertex i
1 $L'' := pred(L')$
2 $\Delta := Q - w(L) - w(L'')$
3 **if** $\Delta < 0$ **then**
4 return `false`;
5 $w_1 := \mathscr{W}(L)$.first()
6 $w_1'' := \mathscr{W}(L'')$.first()
7 **while** $w_1 \neq \mathscr{W}(L).end()$ *and* $w_1'' \neq \mathscr{W}(L'').end()$ **do**
8 $\Delta_1 := Q/2 - w_1 - (w(L'') - w_1'')$
9 **if** $\Delta_1 < 0$ **then**
10 $w_1'' := \mathscr{W}(L'')$.next$(w_1'')$
11 **else**
12 **if** $\Delta_1 \leq \Delta$ **then**
13 return `true`
14 $w_1 := \mathscr{W}(L)$.next(w_1)
15 return `false`

Algorithm 4: Merge Procedure for Compartments of Different Size and $m = 2$

Input : labels L and L' at same vertex i
1 **for** *four cases of Table 1* **do**
2 Modify Steps 6, 7, and 10 in Algorithm 3 for the traversal of $\mathscr{W}(L'')$
3 Modify Step 8 in Algorithm 3 for the computation of Δ_1
4 Call Algorithm 3
5 **if** *result is* `true` **then**
6 return `true`
7 return `false`

Case	Value Δ_1 in Step 8 of Algorithm 3	Traversal of $\mathscr{W}(L'')$
$w_1 + w_2'' \leq Q_1$	$Q_1 - w_1 - (w(L'') - w_1'')$	increasing
$w_1 + w_2'' \leq Q_2$	$Q_2 - w_1 - (w(L'') - w_1'')$	increasing
$w_1 + w_1'' \leq Q_1$	$Q_1 - w_1 - w_1''$	decreasing
$w_1 + w_1'' \leq Q_2$	$Q_2 - w_1 - w_1''$	decreasing

Table 1: Merge Procedure for Compartments of Different Size

one hand, the criterion Δ_1 must be adapted to the specific case. On the other hand, we must ensure that the w_1''-values of $w_1'' \in \mathscr{W}(L'')$ are traversed in decreasing order (as the values of w_2'' in the original version of Algorithm 3). It is here required to initialize w_1'' as the last element of $\mathscr{W}(L'')$ in Step 6, to compare against the reverse end in Step 7, and to move to the preceding element in Step 10. Algorithm 4 calls modified versions of Algorithm 3 up to four times. If all four calls are not successful, i.e., they all return `false`, no overall feasible packing can be determined (Step 7).

In both Algorithms 3 and 4, the run-time complexity is again linearly bounded by $\mathcal{O}(|\mathscr{W}(L^1)|+|\mathscr{W}(L^2)|)$.

4.2.2. The Case of More than Two Compartments

For $m > 2$ compartments, we do *not* rely on sets of truncated packings but use the untruncated packings. We discuss label extension, dominance test, and the merge procedure of bidirectional labeling.

Label Extension. The extension of a label L over an arc $(i, j) \in A$ is a straightforward generalization of the label extension that we have described in Section 4.1. For each $\mathbf{w} \in \mathscr{W}(L)$, the packing $\mathbf{w}$ is treated like a packing in the standard labeling: Identical and different compartment size are handled with different procedures, where the former case allows an ad hoc feasibility test and the latter requires sorting the components of $\mathbf{w}$ first and comparison against $\mathbf{Q}_{\leq}$ second.

Dominance Test. A naive dominance test for two labels L^1 and L^2 could test all pairs $(\mathbf{w}, \mathbf{w}') \in \mathscr{W}(L^1) \times \mathscr{W}(L^2)$ regarding $\mathbf{w} \leq \mathbf{w}'$, see Dominance 2. Such a test has quadratic run time $\mathcal{O}(|\mathscr{W}(L^1)| \cdot |\mathscr{W}(L^2)|)$ and should be avoided whenever possible. Instead, we distinguish two cases comparing the total load of L^1 and L^2.

If the total load is identical, i.e., $w(L^1) = w(L^2)$, dominance can only occur for identical packings $\mathbf{w} = \mathbf{w}'$. Since the sets of packings are lexicographically sorted (the lexicographical ordering is a total ordering), one can identify *all* identical entries in linear time $\mathcal{O}(|\mathscr{W}(L^1)| + |\mathscr{W}(L^2)|)$.

For unequal total load $w(L^1) < w(L^2)$, the worst case complexity remains $\mathcal{O}(|\mathscr{W}(L^1)| \cdot |\mathscr{W}(L^2)|)$, but we observed in pretests that the average-case complexity can be reduced, in particular when the two sets of packings are very unbalanced in size.

Algorithm 5: Partial Dominance for $m \geq 3$ Case $|\mathscr{W}(L^1)| \gg |\mathscr{W}(L^2)|$

```
   Input : labels L^1 and L^2
1  Δ := w(L^2) − w(L^1)
2  w' := W(L^2).first()
3  while w' ≠ W(L^2).end() do
4      next := W(L^2).next(w')
5      w := BinarySearch(W(L^1), w'_1 − Δ)
6      while w_1 ≤ w'_1 do
7          if w ≤ w' then
8              Remove w' from W(L^2)
9              Break
10         w := W(L^1).next(w)
11     w' := next
   Output: (possibly) modified W(L^2)
```

Algorithm 6: Partial Dominance for $m \geq 3$ Case $|\mathscr{W}(L^1)| \ll |\mathscr{W}(L^2)|$

```
   Input : labels L^1 and L^2
1  Δ := w(L^2) − w(L^1)
2  w := W(L^1).first()
3  while w ≠ W(L^1).end() do
4      w := BinarySearch(W(L^2), w_1)
5      while w'_1 ≤ w_1 + Δ do
6          next := W(L^2).next(w')
7          if w ≤ w' then
8              Remove w' from W(L^2)
9          w := next
10     w' := W(L^2).next(w')
   Output: (possibly) modified W(L^2)
```

If $|\mathscr{W}(L^1)| \gg |\mathscr{W}(L^2)|$, one can loop over $\mathbf{w} \in \mathscr{W}(L^2)$ and perform a binary search on $\mathscr{W}(L^1)$ to identify relevant packings $\mathbf{w} \in \mathscr{W}(L^1)$ for the comparison. Algorithm 5 summarizes the refined approach. The outer loop (Steps 3–11) over all packings $\mathbf{w}' \in \mathscr{W}(L^2)$ tests whether $\mathbf{w}'$ is dominated. Since $\mathscr{W}(L^1)$ is lexicographically sorted, we can restrict the potential packing $\mathbf{w} \in \mathscr{W}(L^1)$ to those whose first component is in the interval $[w'_1 - \big(w(L^2) - w(L^1)\big), w'_1]$. Accordingly, the function `BinarySearch` (Step 4) returns the first packing $\mathbf{w}$ from the set $\mathscr{W}(L^1)$ of packings with entry w_1 not smaller than $w'_1 - \big(w(L^2) - w(L^1)\big)$, and the inner loop (Steps 6–10) runs over the packings $\mathbf{w}$ whose first component is in the interval $[w'_1 - \big(w(L^2) - w(L^1)\big), w'_1]$.

If $|\mathscr{W}(L^1)| \ll |\mathscr{W}(L^2)|$, the corresponding procedure is shown in Algorithm 6. Just note that $w_1 \in [w'_1 - \big(w(L^2) - w(L^1)\big), w'_1]$ is equivalent to $w'_1 \in [w_1, w_1 + \big(w(L^2) - w(L^1)\big)]$.

To decide which type of dominance procedure is performed, we use the difference $\delta = |\mathscr{W}(L^1)| - |\mathscr{W}(L^2)|$. We apply the all-pairs comparison for $-20 \leq \delta \leq 20$ and one on the binary search-based tests for $\delta < -20$ and $\delta > 20$, respectively.

Bidirectional Labeling and Merge Procedure. Also the bidirectional labeling is a straightforward generalization of what we described in Section 4.1. When testing a forward label L and a backward label L', we consider the predecessor $L'' = pred(L')$ of L'. We first test the necessary conditions (8). If fulfilled, we perform two nested loops over $\mathbf{w} \in \mathscr{W}(L)$ and over $\mathbf{w}'' \in \mathscr{W}(L'')$, and test whether the two packings $\mathbf{w}$ and $\mathbf{w}''$ can be combined into a feasible packing. As in Section 4.1, the two cases with identical and different compartments are treated specifically.

5. Computational Results

In this section, we describe how we generated the benchmark instances, we present details of the BPC implementation and parameterization and the computational setting, and we report computational results. A comparison between instances with identical and different compartment size closes the section.

5.1. Benchmark Instances

To compare the two types of BPC algorithms, we generated BMCVRP instances as follows. The locations of the depot and customers are discretely uniformly distributed on a $[1, 100] \times [1, 100]$ grid. We consider vertex sets $V = \{0, 1, 2, \ldots, n-1, n\}$ with $n \in \{10, 20, \ldots, 100\}$. The number m of compartments is 2, 3, or 4. To compare the impact of shorter and longer route lengths on the algorithms' performance, the vehicle capacities are varied with $Q = 120$ or $Q = 240$, and the demand d_i is discretely uniformly distributed on $[1, Q/4] \cap \mathbb{N}$ for $Q = 120$ and on $[1, Q/6] \cap \mathbb{N}$ for $Q = 240$, respectively. Moreover, we consider instances with identical and different compartment size as specified in Table 2. We generated five instances of each kind yielding 600 instances in total. The instances are online available at the online archive (iRODS, 2021).

Table 2: Overview of compartment sizes $\mathbf{Q} = (Q_1, \ldots, Q_m)$ with total capacity $Q = \sum_{k=1}^{m} Q_k$.

	Total Capacity $Q = 120$		Total Capacity $Q = 240$	
Compartments	identical	different	identical	different
$m = 2$	$(60, 60)$	$(40, 80)$	$(120, 120)$	$(80, 160)$
$m = 3$	$(40, 40, 40)$	$(20, 40, 60)$	$(80, 80, 80)$	$(40, 80, 120)$
$m = 4$	$(30, 30, 30, 30)$	$(10, 15, 30, 65)$	$(60, 60, 60, 60)$	$(20, 30, 60, 130)$

5.2. Branch-Price-and-Cut Setup

Both BPC algorithms, the one that uses the standard labeling and the one with partial dominance, are configured identically regarding the general strategies for pricing, cutting, and branching. We briefly summarize the parameterization of these algorithmic components.

Pricing Strategy. The *ng*-route relaxation defining the subproblem to be solved in the pricing subproblem uses neighborhoods N_i of size 14 for all $i \in N$. The neighborhood N_i contains the customer i and the 13 closest customers to i. As a fast heuristic for partial pricing (Gamache *et al.*, 1999), we use nearest neighbor networks of increasing size with 2, 5, 10, and 15 neighbors before the complete network is explored.

Cutting Strategy. We restrict the addition of CCs and SRIs to branch-and-bound nodes up to level 3 (root node, the two child nodes, and the four grand children). A maximum of 300 violated CCs is added. They are identified by the greedy shrinking heuristic and the extended shrinking heuristic presented by Ralphs *et al.* (2003). The separation procedure for SRIs is only employed when no violated CC has been found. We use the refined version of SRIs with a limited memory as described in (Pecin *et al.*, 2017). Per round of separation, we identify via enumeration the 10 most violated SRIs defined by a row subset S with $|S| = 3$ requiring a minimum violation of 0.05. Up to 320 SRIs are added in total.

Branching Strategy. We apply the standard two-level branching strategy: On the first level, we create two branches $f \leq \lfloor \bar{f} \rfloor$ and $f \geq \lfloor \bar{f} \rfloor + 1$ if the number $\bar{f} = \sum_{r \in \Omega} \bar{\lambda}^r$ of employed vehicles is fractional (note that we distinguish between values $\bar{f}$ and $\bar{\lambda}^r$ and decision variables f and λ^r).

On the second level, we branch on the overall flow on edges $ij \in E$ for customers $i, j \in N$, if $\bar{x}_{ij} = \sum_{r \in \Omega} (h^r_{ij} + h^r_{ji}) \bar{\lambda}^r$ is fractional. In the zero-branch, the two associated arcs (i, j) and (j, i) are eliminated from the SPPRC digraph, while in the one-branch, we add the constraint $\sum_{r \in \Omega} (h^r_{ij} + h^r_{ji}) \lambda^r \geq 1$ to the RMP. We use strong branching (Achterberg *et al.*, 2005) where candidate edges ij are those with a value $\bar{x}_{ij}$ closest to $1/2$. For each candidate edge, the two child nodes are solved and the best edge is determined with the help of the product rule of Achterberg *et al.* We evaluate up to 8 different edges at the root node, decrease the number of candidates so that at the level 10 of the branch-and-bound tree only two candidates are evaluated. Deeper in the search tree, strong branching is turned off, i.e., one edge with value $\bar{x}_{ij}$ closest to $1/2$ is directly chosen.

5.3. Computational Setup

The algorithms are implemented in C++ and compiled into 64-bit single-thread code with Microsoft Visual Studio 2015. CPLEX 12.10.0 is utilized to solve the RMP at each column-generation iteration. Moreover, CPLEX is used as a primal MIP-based heuristic solver after the solution of each branch-and-bound node using all generated columns. Apart from setting the number of threads to 1, CPLEX's default values are kept for all other parameters. The computational study is carried out on a 64-bit Microsoft Windows 10 computer equipped with an Intel® Core™ i7-5930k CPU clocked at 3.5 GHz and 64 GB of RAM. Computation times are limited to a maximum of 3600 seconds per instance.

5.4. Results

Tables 3 and 4, for instances with vehicle capacity $Q = 120$ and $Q = 240$, respectively, summarize the comparison of the BPC algorithms using standard labeling (including the refinements presented in Section 4.1) and labeling with partial dominance. Additionally, the Appendix contains detailed results per instance for the BCP algorithm with partial dominance. The table entries of the Tables 3 and 4 have the following meaning:

#inst:	number of instances;
#opt:	number of instances solved to proven optimality within 1 hour (3600 seconds);
time $\bar{t}$:	average computation time in seconds; unsolved instances are taken into account with the time limit TL of 1 hour (3600 seconds);
gap:	$100 \cdot (UB - LB)/LB$, i.e., the average gap in percent at termination (the average is taken only over the instances for which an upper bound was computed with the BPC algorithm).

The BPC algorithm using partial dominance can solve many more instances to proven optimality and has a lower average computation time compared to the one with standard labeling. For the smaller capacity of $Q = 120$, 83 of 150 instances with identical compartment size and 72 of 150 instances with different compartment size can be solved exactly. Likewise, for the higher capacity of $Q = 240$ and identical (different) compartment size, 63 (58) instances can be solved by utilizing partial dominance and only 19 (19) by utilizing standard labeling. The average computation time is reduced by approximately 20% to 30% on average (depending on the group of instances) when utilizing partial dominance for instances with identical or different compartment size, respectively. Comparing both algorithms on the subset of better solvable instances with $|V| \leq 40$, the average computation time is even one or two orders of magnitudes smaller. Overall, the version with partial dominance is superior compared to the algorithm utilizing standard labeling, which is also confirmed by the performance profile (Dolan and Moré, 2002) shown in Figure 1. Note that for a set of algorithms $\mathcal{A}$, the performance profile $\rho_A(\tau)$ of an algorithm $A \in \mathcal{A}$ describes the ratio of instances that can be solved by A within a factor τ compared to the fastest algorithm, i.e., $\rho_A(\tau) = \left|\left\{I \in \mathcal{I} : t_I^A / t_I^* \leq \tau\right\}\right| / |\mathcal{I}|$.

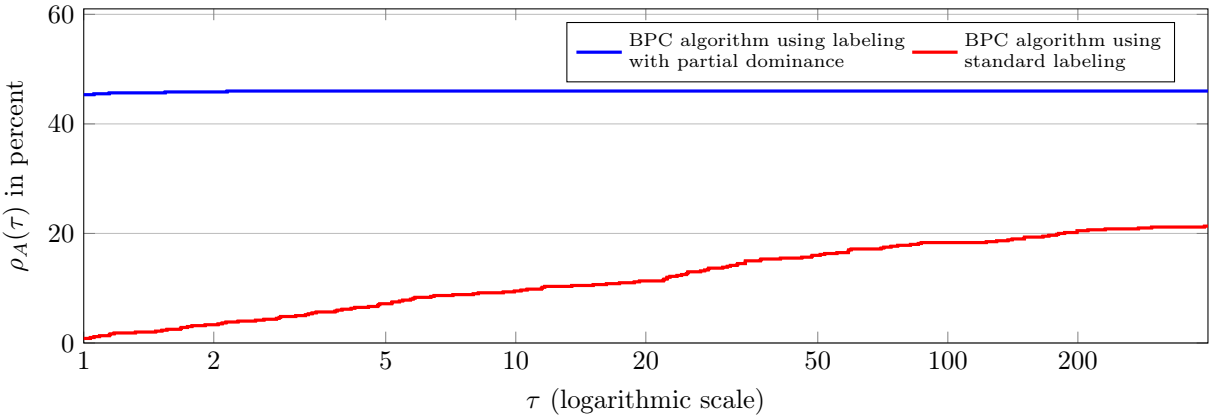

Figure 1: Performance profile comparing the BPC algorithm that uses labeling with partial dominance and the BPC algorithm that uses standard labeling.

We highlight some additional findings: Apart from a few exceptions, instances with more vertices and more compartments are more difficult to solve. Moreover, regardless of the number m of compartments, the number of optimally solved instances with identical compartments is higher in comparison to instances with different compartment size. This can be attributed to the faster algorithmic components in the labeling, i.e., label extension, dominance, and merge procedure (cf. Section 4). In comparison, more instances with capacity $Q = 120$ are solved compared to instances with $Q = 240$.

Table 3: Results for instances with vehicle capacity $Q = 120$.

			Identical compartments						Different compartment size					
			Standard labeling			Partial dominance			Standard labeling			Partial dominance		
m	$\|V\|$	#inst	#opt	time $\bar{t}$	gap	#opt	time $\bar{t}$	gap	#opt	time $\bar{t}$	gap	#opt	time $\bar{t}$	gap
2	10	5	**5**	**0.2**	0.0	**5**	**0.2**	0.0	**5**	0.3	0.0	**5**	**0.2**	0.0
	20	5	**5**	306.4	0.0	**5**	**25.3**	0.0	**5**	141.1	0.0	**5**	**17.0**	0.0
	30	5	**5**	285.2	0.0	**5**	**5.1**	0.0	4	1022.5	0.3	**5**	**2.3**	0.0
	40	5	4	781.8	0.0	**5**	**27.4**	0.0	4	905.1	0.0	**5**	**30.2**	0.0
	50	5	2	2799.0	1.7	**3**	**2119.0**	1.6	1	3221.6	0.1	**2**	**2183.3**	1.8
	60	5	1	3231.3	0.0	**5**	**1539.6**	0.0	0	*TL*	0.1	**3**	**2259.1**	0.0
	70	5	0	*TL*		**2**	**2685.3**	0.5	0	*TL*		**2**	**3196.3**	0.4
	80	5	0	*TL*		0	*TL*	0.9	0	*TL*		**1**	**3234.3**	0.8
	90	5	0	*TL*		**1**	**3286.6**	0.7	0	*TL*		0	*TL*	0.9
	100	5	0	*TL*		0	*TL*	1.1	0	*TL*		0	*TL*	1.0
Subtotal		50	22	2180.4	0.3	**31**	**1688.8**	0.5	19	2329.1	0.1	**28**	**1812.3**	0.5
3	10	5	**5**	0.6	0.0	**5**	**0.2**	0.0	**5**	2.4	0.0	**5**	**0.1**	0.0
	20	5	**5**	572.3	0.0	**5**	**3.3**	0.0	4	827.9	0.0	**5**	**4.3**	0.0
	30	5	4	1135.6	0.0	**5**	**176.6**	0.0	3	1553.6	0.0	**5**	**39.4**	0.0
	40	5	3	2098.8	0.0	**5**	**100.1**	0.0	2	2938.9	0.2	**5**	**282.6**	0.0
	50	5	0	*TL*	1.4	**3**	**1741.8**	1.2	0	*TL*		**2**	**2520.1**	0.8
	60	5	0	*TL*		0	*TL*	0.5	0	*TL*		**1**	**3052.0**	0.4
	70	5	0	*TL*		**2**	**3024.9**	0.1	0	*TL*		**1**	**3507.1**	0.0
	80	5	0	*TL*		**1**	**2939.2**	0.5	0	*TL*		0	*TL*	1.3
	90	5	0	*TL*		0	*TL*	1.0	0	*TL*		0	*TL*	
	100	5	0	*TL*		**1**	**3321.2**	0.0	0	*TL*		0	*TL*	
Subtotal		50	17	2540.7	0.1	**27**	**1850.7**	0.3	14	2692.3	0.0	**24**	**2020.6**	0.3
4	10	5	**5**	0.4	0.0	**5**	**0.1**	0.0	**5**	23.3	0.0	**5**	**0.1**	0.0
	20	5	4	740.5	0.0	**5**	**4.3**	0.0	3	1649.9	0.0	**5**	**26.6**	0.0
	30	5	4	1429.0	0.0	**5**	**12.8**	0.0	3	1610.0	0.0	**4**	**765.4**	0.0
	40	5	1	3437.3	0.0	**5**	**734.6**	0.0	0	*TL*		**4**	**1174.4**	0.0
	50	5	0	*TL*	1.0	**1**	**2891.2**	0.3	0	*TL*		**1**	**3302.8**	1.5
	60	5	0	*TL*		**2**	**2390.2**	0.3	0	*TL*		0	*TL*	0.4
	70	5	0	*TL*		**1**	**2944.8**	0.2	0	*TL*		0	*TL*	0.9
	80	5	0	*TL*		**1**	**3267.2**	0.4	0	*TL*		**1**	**3235.1**	0.2
	90	5	0	*TL*		0	*TL*	0.4	0	*TL*		0	*TL*	
	100	5	0	*TL*		0	*TL*	0.2	0	*TL*		0	*TL*	
Subtotal		50	14	2600.8	0.1	**25**	**1944.5**	0.1	11	2745.8	0.0	**20**	**2290.4**	0.3
Total		150	53	2434.0	0.2	**83**	**1828.0**	0.3	44	2582.5	0.1	**72**	**2041.1**	0.4

Note: Larger numbers of exactly solved instances and smaller average times are highlighted in bold.

Summarizing, the most difficult BMCVRP instances of our test set are those with $m = 4$ compartments of different size that provide a larger total capacity of $Q = 240$. Here, the BPC algorithm with the standard labeling completely fails for all instances with more than ten customers, while the one with partial dominance still optimally solves all instances with 20 and two of the five instances with 30 customers.

As a side note we mention that the computation of the tight fleet-size lower bound $F^{\star}_{LB}$ (see Section 2.2) never gave a strictly larger number than the trivial lower bound $\lceil \sum_{i \in N} d_i / Q \rceil$ for any of the 600 test instances. One can provoke different lower bounds by increasing the average demand (for a discussion and analysis of demand distributions and their impact on the employed fleets and route length in the context of split-delivery VRPTWs, see Bianchessi *et al.*, 2019, Sections 1 and 5), but the resulting BMCVRP instances

Table 4: Results for instances with vehicle capacity $Q = 240$.

			Identical compartments						Different compartment size					
			Standard labeling			Partial dominance			Standard labeling			Partial dominance		
m	$\|V\|$	#inst	#opt	time $\bar{t}$	gap	#opt	time $\bar{t}$	gap	#opt	time $\bar{t}$	gap	#opt	time $\bar{t}$	gap
2	10	5	**5**	0.3	0.0	**5**	**0.1**	0.0	**5**	0.5	0.0	**5**	**0.1**	0.0
	20	5	4	1321.2	0.0	**5**	**4.3**	0.0	4	1718.6	0.0	**5**	**1.4**	0.0
	30	5	0	*TL*		**5**	**87.2**	0.0	0	*TL*		**5**	**76.3**	0.0
	40	5	0	*TL*		**4**	**756.7**	0.0	0	*TL*		**4**	**765.4**	0.0
	50	5	0	*TL*		**4**	**856.9**	1.8	0	*TL*		**4**	**873.6**	2.0
	60	5	0	*TL*		**2**	**2791.2**	0.6	0	*TL*		**2**	**2754.9**	0.3
	70	5	0	*TL*		**2**	**2964.7**	0.0	0	*TL*		**1**	**3052.4**	0.0
	80	5	0	*TL*		0	*TL*	0.6	0	*TL*		0	*TL*	0.9
	90	5	0	*TL*		0	*TL*	2.8	0	*TL*		0	*TL*	3.3
	100	5	0	*TL*		0	*TL*	3.8	0	*TL*		0	*TL*	
Subtotal		50	9	3012.2	0.0	**27**	**1826.1**	0.6	9	3051.9	0.0	**26**	**1832.4**	0.5
3	10	5	**5**	1.7	0.0	**5**	**< 0.1**	0.0	**5**	7.6	0.0	**5**	**0.1**	0.0
	20	5	0	*TL*		**5**	**14.9**	0.0	0	*TL*		**5**	**23.4**	0.0
	30	5	0	*TL*		**5**	**768.6**	0.0	0	*TL*		**4**	**1110.0**	0.0
	40	5	0	*TL*		**4**	**1276.8**	0.0	0	*TL*		**4**	**1752.9**	0.0
	50	5	0	*TL*		**2**	**2569.2**	0.0	0	*TL*		**2**	**2410.7**	0.0
	60	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	70	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	80	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	90	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	100	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
Subtotal		50	5	3240.2	0.0	**21**	**2263.0**	0.0	5	3240.8	0.0	**20**	**2329.7**	0.0
4	10	5	**5**	2.0	0.0	**5**	**0.1**	0.0	**5**	48.3	0.0	**5**	**0.4**	0.0
	20	5	0	*TL*		**5**	**74.6**	0.0	0	*TL*		**5**	**605.3**	0.0
	30	5	0	*TL*		**3**	**2429.3**	0.0	0	*TL*		**2**	**2917.7**	0.0
	40	5	0	*TL*		**2**	**2576.0**	0.0	0	*TL*		0	*TL*	
	50	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	60	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	70	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	80	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	90	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
	100	5	0	*TL*		0	*TL*		0	*TL*		0	*TL*	
Subtotal		50	5	3171.7	0.0	**15**	**2668.0**	0.0	5	3177.2	0.0	**12**	**2872.3**	0.0
Total		150	19	3139.6	0.0	**63**	**2252.4**	0.3	19	3155.5	0.0	**58**	**2344.8**	0.2

Note: Larger numbers of exactly solved instances and smaller average times are highlighted in bold.

then become much simpler to solve. Moreover, we analyzed the computed 276 (of 600) optimal solutions. We found that 236 solutions utilize exactly $F^{\star}_{LB}$ vehicles, 37 utilize exactly $F^{\star}_{LB} + 1$ vehicles, and three utilize $F^{\star}_{LB} + 2$ vehicles.

5.5. Comparison for Identical and Different Compartment Size

In this section, we compare optimal solutions for instances with identical compartment size with those that have different compartment size. For each BMCVRP instance with identical compartments, there is one corresponding instance with compartments of different size but otherwise identical characteristics, i.e., identical depot and customers, identical total capacity Q, and identical number of compartments m. We

Table 5: Comparison between instances with identical (*idt*) compartments and compartments of different size (*diff*).

		Count z_{idt} versus z_{diff}				Cost	
m	$\|V\|$	#opt both	$\neq$	$>$	$<$	$\bar{z}_{idt}$	$\bar{z}_{diff}$
2	10	10	0	0	0	3214.9	3214.9
	20	10	0	0	0	4852.6	4852.6
	30	10	2	2	0	6297.9	**6293.0**
	40	9	1	1	0	7668.8	**7667.9**
	50	6	1	1	0	8545.8	**8537.2**
	60	5	2	1	1	**9626.6**	9635.2
	70	3	1	1	0	10649.0	**10646.3**
Subtotal		53	7	6	1	6491.1	**6489.7**
3	10	10	2	**2**	0	3270.6	**3214.9**
	20	10	4	4	0	4992.3	**4914.5**
	30	9	4	4	0	6496.9	**6428.2**
	40	9	5	5	0	7830.8	**7746.6**
	50	4	3	3	0	9244.0	**9068.5**
Subtotal		42	18	18	0	5918.0	**5836.7**
4	10	10	2	1	1	**3234.0**	3248.1
	20	10	4	1	3	**4916.4**	4956.7
	30	6	2	0	2	**6655.3**	6740.0
	40	4	3	2	1	8536.3	**8527.8**
	50	1	1	1	0	14265.0	**14019.0**
Subtotal		31	12	5	7	**5478.9**	5503.8
Total		126	37	29	8	6051.0	**6029.5**

Note: Smaller costs are highlighted in bold.

use the superior BPC algorithm with partial dominance to compute optimal solutions. For the comparison, we only consider pairs of instances that are optimally solved for both compartment settings. Results are summarized in Table 5, where the table entries have the following meaning:

#opt both: number of pairs of corresponding instances solved to proven optimality within 1 hour (3600 seconds);
$\neq$: number of instances with different total cost, i.e., $z_{idt} \neq z_{diff}$;
$>$: number of instances with $z_{idt} > z_{diff}$;
$<$: number of instances with $z_{idt} < z_{diff}$;
$\bar{z}_{diff}$: average optimal objective value for instances with different compartment size;
$\bar{z}_{idt}$: average optimal objective value for instances with identical compartment size.

For instances with only $m = 2$ compartments and up to 20 customers, we observe no differences in optimal solutions for identical and different compartment sizes. However, starting from $m \geq 3$ or $|V| \geq 30$, at least one pair of corresponding instances shows a difference. For all numbers m of compartments, the proportion of instances with different optima increases with the number of vertices $|V|$. For $|V| \geq 40$, different costs occur in 40% of the cases.

In most of the cases, namely 29 out of 37, the cost of solutions to instances with different compartment size is lower than the cost of those with identical compartment size. We explain this observation with the fact that one larger compartment is generally beneficial as occurring in instances with different compartment sizes (see Table 2). The larger compartment provides more flexibility for packing items. However, if the smaller

compartments become too small, as it happens for $m = 4$, the average total cost of different compartment size is higher than the cost of identical compartments. Overall, the average relative difference in cost is less than 0.1%, i.e., very small.

By construction (see Table 2), instances with $m = 2$ compartments are relaxations of instances with $m = 4$ compartments, i.e., every feasible solution for $m = 4$ is also feasible for the corresponding instance with $m = 2$. The cost comparison in Table 5 reflects this fact when comparing one row for $m = 2$ with the corresponding row (i.e., identical number of customers) for $m = 4$. Note however that cost averages are taken over samples of different size so that opposite cost relations could also occur.

6. Conclusions

Dominance rules are of high importance for the effectiveness of SPPRC labeling algorithms. Many studies confirm that the largest share of the computation time of a BPC algorithm in vehicle and crew routing and scheduling is spent in the SPPRC subproblems. Within the SPPRC labeling algorithm, the dominance procedure is often the critical part and responsible for the consumption of more than 90% of the entire BPC computation time.

In this work, we have introduced a new type of partial dominance tailored to the BMCVRP, a VRP variant that synthesizes vehicle routing and one-dimensional bin packing with possibly multiple bin sizes. For the BMCVRP, the new SPPRC labeling algorithm with partial dominance has been shown striking performance compared to standard labeling. The properties that make labeling with partial dominance superior can be attributed to the following facts: Labeling with partial dominance

- eliminates identical packings immediately during the extension step (in standard labeling, the elimination is postponed to the dominance test, which leads to the generation of more labels that are soon afterwards eliminated again);
- represents many possible packings within a label and, thus, avoids the same comparisons of the cost, *ng*-route, and SRI attributes in the dominance algorithm;
- also avoids identical calculations and feasibility checks of the above attributes in the merge procedure;
- reduces redundancy in the merge process (in standard labeling, it is hardly possible to directly avoid the generation of identical routes that only differ in the packings; these are typically filtered out in a final step of the labeling algorithm).

Compared to published approaches that also introduced versions of partial dominance, the new BMCVRP-tailored partial dominance has different nature: Older approaches exploited partial dominance either to solve a relaxation of the elementary SPPRC or to cope with a continuous tradeoff between two SPPRC attributes (see discussion of the literature in the introduction). The partial dominance for the BMCVRP is of discrete nature, because finite sets of packings of the compartments are thinned out.

We encourage other researchers to try out partial dominance in their dynamic-programming algorithms. The presented results suggest that the principle of partial dominance has the potential to substantially accelerate the solution of some SPPRC subproblems, but it may also help in very different problems solved with dynamic programming.

Acknowledgement

This research was supported by the Deutsche Forschungsgemeinschaft (DFG) under grant IR 122/10-1. This support is gratefully acknowledged.

References

Achterberg, T., Koch, T., and Martin, A. (2005). Branching rules revisited. *Operations Research Letters*, **33**(1), 42–54.

Agra, A., Christiansen, M., and Delgado, A. (2013). Mixed integer formulations for a short sea fuel oil distribution problem. *Transportation Science*, **47**(1), 108–124.

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

Bianchessi, N., Drexl, M., and Irnich, S. (2019). The split delivery vehicle routing problem with time windows and customer inconvenience constraints. *Transportation Science*, **53**(4), 1067–1084.

Bode, C. and Irnich, S. (2012). Cut-first branch-and-price-second for the capacitated arc-routing problem. *Operations Research*, **60**(5), 1167–1182.

Bulhões, T., Sadykov, R., and Uchoa, E. (2018). A branch-and-price algorithm for the minimum latency problem. *Computers & Operations Research*, **93**, 66–78.

Cherkesly, M. and Gschwind, T. (2020). The pickup and delivery problem with time windows, multiple stacks, and handling operations. Technical report, Les Cahiers du GERAD G–2020–16, GERAD, HEC Montréal, Canada.

Cherkesly, M., Desaulniers, G., Irnich, S., and Laporte, G. (2016). Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and multiple stacks. *European Journal of Operational Research*, **250**(3), 782–793.

Christiansen, M., Fagerholt, K., Flatberg, T., Haugen, Ø., Kloster, O., and Lund, E. H. (2011). Maritime inventory routing with multiple products: A case study from the cement industry. *European Journal of Operational Research*, **208**(1), 86–94.

Coelho, L. C. and Laporte, G. (2015). Classification, models and exact algorithms for multi-compartment delivery problems. *European Journal of Operational Research*, **242**(3), 854–864.

Cornillier, F., Boctor, F. F., Laporte, G., and Renaud, J. (2008). An exact algorithm for the petrol station replenishment problem. *Journal of the Operational Research Society*, **59**(5), 607–615.

Correia, I., Gouveia, L., and Saldanha-da Gama, F. (2008). Solving the variable size bin packing problem with discretized formulations. *Computers & Operations Research*, **35**(6), 2103–2113.

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

Delorme, M. and Iori, M. (2020). Enhanced pseudo-polynomial formulations for bin packing and cutting stock problems. *INFORMS Journal on Computing*, **32**(1), 101–119.

Desaulniers, G. (2010). Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations Research*, **58**(1), 179–192.

Desaulniers, G. and Villeneuve, D. (2000). The shortest path problem with time windows and linear waiting costs. *Transportation Science*, **34**(3), 312–319.

Desaulniers, G., Desrosiers, J., Ioachim, I., M. Solomon, M., Soumis, F., and Villeneuve, D. (1998). A unified framework for deterministic time constrained vehicle routing and crew scheduling problems. In T. G. Crainic and G. Laporte, editors, *Fleet Management and Logistics*, pages 57–93. Springer.

Desaulniers, G., Desrosiers, J., and Solomon, M., editors (2005). *Column Generation*. Springer, New York.

Desaulniers, G., Rakke, J. G., and Coelho, L. C. (2016a). A branch-price-and-cut algorithm for the inventory-routing problem. *Transportation Science*, **50**(3), 1060–1076.

Desaulniers, G., Errico, F., Irnich, S., and Schneider, M. (2016b). Exact algorithms for electric vehicle-routing problems with time windows. *Operations Research*, **64**(6), 1388–1405.

Desaulniers, G., Gschwind, T., and Irnich, S. (2020). Variable fixing for two-arc sequences in branch-price-and-cut algorithms on path-based models. *Transportation Science*, **54**(5), 1170–1188.

Dolan, E. D. and Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming*, **91**(2), 201–213.

Drexl, M. (2007). *On some generalized routing problems*. Ph.d. dissertation, Faculty of Business and Economics, RWTH Aachen University, Aachen, Germany. http://publications.rwth-aachen.de/record/62536.

Gamache, M., Soumis, F., Marquis, G., and Desrosiers, J. (1999). A column generation approach for large-scale aircrew rostering problems. *Operations Research*, **47**(2), 247–263.

Goeke, D., Gschwind, T., and Schneider, M. (2019). Upper and lower bounds for the vehicle-routing problem with private fleet and common carrier. *Discrete Applied Mathematics*, **264**, 43–61.

Gschwind, T. and Irnich, S. (2015). Effective handling of dynamic time windows and its application to solving the dial-a-ride problem. *Transportation Science*, **49**(2), 335–354.

Gschwind, T., Bianchessi, N., and Irnich, S. (2019). Stabilized branch-price-and-cut for the commodity-constrained split delivery vehicle routing problem. *European Journal of Operational Research*, **278**(1), 91–104.

He, Q., Irnich, S., and Song, Y. (2019). Branch-and-cut-and-price for the vehicle routing problem with time windows and convex node costs. *Transportation Science*, **53**(5), 1409–1426.

Henke, T., Speranza, M. G., and Wäscher, G. (2015). The multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research*, **246**(3), 730–743.

Heßler, K. (2021). Exact algorithms for the multi-compartment vehicle routing problem with flexible compartment sizes. *European Journal of Operational Research*, **294**(1), 188–205.

Heßler, K., Irnich, S., Kreiter, T., and Pferschy, U. (2021). Bin packing with lexicographic objectives for loading weight- and volume-constrained trucks in a direct-shipping system. *OR Spectrum*. doi: 10.1007/s00291-021-00628-x.

Houck, D., Picard, J., Queyranne, M., and Vemuganti, R. (1980). The travelling salesman problem as a constrained shortest path problem: theory and computational experience. *Opsearch*, **17**, 93–109.

Ioachim, I., Gélinas, S., Desrosiers, J., and Soumis, F. (1998). A dynamic programming algorithm for the shortest path problem with time windows and linear node costs. *Networks*, **31**, 193–204.

Ioachim, I., Desrosiers, J., Soumis, F., and Bélanger, N. (1999). Fleet assignment and routing with schedule synchronization constraints. *European Journal of Operational Research*, **119**(1), 75–90.

Irnich, S. and Desaulniers, G. (2005). Shortest path problems with resource constraints. In G. Desaulniers, J. Desrosiers, and M. M. Solomon, editors, *Column Generation*, chapter 2, pages 33–65. Springer.

Irnich, S. and Villeneuve, D. (2006). The shortest path problem with resource constraints and k-cycle elimination for $k \geq 3$.

INFORMS Journal on Computing, **18**(3), 391–406.

iRODS (2021). BMCVRP benchmark instances. iRODS: integrated Rule-Oriented Data System. ZDV, Johannes Gutenberg University Mainz. https://irods-web.zdv.uni-mainz.de/irods-rest/rest/fileContents/zdv/home/khessler/instances/instances_BMCVRP.zip?ticket=JiT8YK5QQSOKMkW.

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

Kiilerich, L. and Wøhlk, S. (2017). New large-scale data instances for CARP and new variations of CARP. *INFOR: Information Systems and Operational Research*, **56**(1), 1–32.

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

Lahyani, R., Coelho, L. C., Khemakhem, M., Laporte, G., and Semet, F. (2015). A multi-compartment vehicle routing problem arising in the collection of olive oil in Tunisia. *Omega*, **51**, 1–10.

Liberatore, F., Righini, G., and Salani, M. (2010). A column generation algorithm for the vehicle routing problem with soft time windows. *4OR*, **9**(1), 49–82.

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

Oppen, J. and Løkketangen, A. (2008). A tabu search approach for the livestock collection problem. *Computers & Operations Research*, **35**(10), 3213–3229.

Ostermeier, M., Martins, S., Amorim, P., and Hübner, A. (2018). Loading constraints for a multi-compartment vehicle routing problem. *OR Spectrum*, **40**(4), 997–1027.

Ostermeier, M., Henke, T., Hübner, A., and Wäscher, G. (2021). Multi-compartment vehicle routing problems: State-of-the-art, modeling framework and future directions. *European Journal of Operational Research*, **292**(3), 799–817.

Parragh, S. N. and Cordeau, J.-F. (2017). Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows. *Computers & Operations Research*, **83**, 28–44.

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

Ralphs, T., Kopman, L., Pulleyblank, W., and Trotter, L. (2003). On the capacitated vehicle routing problem. *Mathematical programming*, **94**(2-3), 343–359.

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

Rothenbächer, A.-K., Drexl, M., and Irnich, S. (2018). Branch-and-price-and-cut for the truck-and-trailer routing problem with time windows. *Transportation Science*, **52**(5), 1174–1190.

Spliet, R. and Gabor, A. F. (2015). The time window assignment vehicle routing problem. *Transportation Science*, **49**(4), 721–731.

Tilk, C., Rothenbächer, A.-K., Gschwind, T., and Irnich, S. (2017). Asymmetry matters: Dynamic half-way points in bidirectional labeling for solving shortest path problems with resource constraints faster. *European Journal of Operational Research*, **261**(2), 530–539.

Tilk, C., Bianchessi, N., Drexl, M., Irnich, S., and Meisel, F. (2018). Branch-and-price-and-cut for the active-passive vehicle-routing problem. *Transportation Science*, **52**(2), 300–319.

Toth, P. and Vigo, D., editors (2014). *Vehicle Routing: Problems, Methods, and Applications*. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics, Philadelphia.

Valerió de Carvalho, J. M. (1998). Exact solution of cutting stock problems using column generation and branch-and-bound. *International Transactions in Operational Research*, **5**(1), 35–44.

Appendix

In this Appendix, we present instance-by-instance results. The entries in Table 6 have the following meaning:

No.:	number of the instance;
$\|V\|$:	number of vertices;
Q:	total capacity;
m:	number of compartments;
$\mathbf{Q}$:	capacity vector;
opt:	an asterisk * indicates if the respective algorithm could solve the instance to proven optimality within 1 hour of computation time;
UB:	upper bound;
LB_{tree}:	lower bound when reaching the time limit of 1 hour;
LB_{LP}:	linear relaxation lower bound;
LB_{cut}:	linear relaxation lower bound of restricted master problem with cutting planes;
gap:	percentage optimality gap $100 \cdot (UB - LB_{\text{tree}})/LB_{\text{tree}}$ at termination;
time:	computation time in seconds; TL indicates that the time limit of 1 hour was reached;
#BaB:	number of solved branch-and-bound nodes;
#CC:	number of capacity cuts added;
#SRI:	number of subset-row inequalities added.

Table 6: Detailed results for all instances solved by labeling with partial dominance.

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
1	10	120	2	(60, 60)	*	408.3	408.3	344.1	408.3	0.0	0.5	1	300	6
2	10	120	2	(40, 80)	*	408.3	408.3	344.1	408.3	0.0	0.4	1	300	6
3	10	120	3	(40, 40, 40)	*	408.3	408.3	344.1	408.3	0.0	0.4	1	300	6
4	10	120	3	(20, 40, 60)	*	408.3	408.3	344.1	408.3	0.0	0.4	1	300	6
5	10	120	4	(30, 30, 30, 30)	*	408.3	408.3	358.4	408.3	0.0	0.4	1	300	6
6	10	120	4	(10, 15, 30, 65)	*	408.3	408.3	349.9	408.3	0.0	0.1	1	19	0
7	10	120	2	(60, 60)	*	361.3	361.3	331.6	361.3	0.0	0.1	1	20	0
8	10	120	2	(40, 80)	*	361.3	361.3	331.6	361.3	0.0	0.1	1	19	0
9	10	120	3	(40, 40, 40)	*	361.3	361.3	332.1	361.3	0.0	<0.1	1	16	0
10	10	120	3	(20, 40, 60)	*	361.3	361.3	331.6	361.3	0.0	0.1	1	19	0
11	10	120	4	(30, 30, 30, 30)	*	361.3	361.3	342.9	361.3	0.0	<0.1	1	13	0
12	10	120	4	(10, 15, 30, 65)	*	361.3	361.3	342.9	361.3	0.0	0.1	1	13	0
13	10	120	2	(60, 60)	*	300.0	300.0	300.0	300.0	0.0	0.1	1	0	0
14	10	120	2	(40, 80)	*	300.0	300.0	300.0	300.0	0.0	0.1	1	0	0
15	10	120	3	(40, 40, 40)	*	300.0	300.0	300.0	300.0	0.0	<0.1	1	0	0
16	10	120	3	(20, 40, 60)	*	300.0	300.0	300.0	300.0	0.0	<0.1	1	0	0
17	10	120	4	(30, 30, 30, 30)	*	300.0	300.0	300.0	300.0	0.0	0.1	1	0	0
18	10	120	4	(10, 15, 30, 65)	*	300.0	300.0	300.0	300.0	0.0	0.1	1	0	0
19	10	120	2	(60, 60)	*	299.2	299.2	292.1	299.2	0.0	0.1	1	22	0
20	10	120	2	(40, 80)	*	299.2	299.2	292.1	299.2	0.0	0.1	1	22	0
21	10	120	3	(40, 40, 40)	*	321.7	321.7	310.7	321.7	0.0	0.3	1	300	9
22	10	120	3	(20, 40, 60)	*	299.2	299.2	292.1	299.2	0.0	0.1	1	23	0
23	10	120	4	(30, 30, 30, 30)	*	306.8	306.8	299.8	306.8	0.0	<0.1	1	21	0
24	10	120	4	(10, 15, 30, 65)	*	299.2	299.2	292.1	299.2	0.0	0.1	1	25	0
25	10	120	2	(60, 60)	*	374.3	374.3	364.5	374.3	0.0	0.1	1	17	0
26	10	120	2	(40, 80)	*	374.3	374.3	361.8	374.3	0.0	0.1	1	35	0
27	10	120	3	(40, 40, 40)	*	407.5	407.5	407.5	407.5	0.0	<0.1	1	0	0
28	10	120	3	(20, 40, 60)	*	374.3	374.3	374.3	374.3	0.0	<0.1	1	0	0
29	10	120	4	(30, 30, 30, 30)	*	385.8	385.8	385.8	385.8	0.0	<0.1	1	0	0
30	10	120	4	(10, 15, 30, 65)	*	407.5	407.5	407.5	407.5	0.0	<0.1	1	0	0
31	10	240	2	(120, 120)	*	336.4	336.4	336.4	336.4	0.0	0.1	1	0	0
32	10	240	2	(80, 160)	*	336.4	336.4	336.4	336.4	0.0	0.1	1	0	0
33	10	240	3	(80, 80, 80)	*	336.4	336.4	336.4	336.4	0.0	0.1	1	0	0
34	10	240	3	(40, 80, 120)	*	336.4	336.4	336.4	336.4	0.0	0.1	1	0	0
35	10	240	4	(60, 60, 60, 60)	*	336.4	336.4	336.4	336.4	0.0	0.1	1	0	0
36	10	240	4	(20, 30, 60, 130)	*	336.4	336.4	336.4	336.4	0.0	1.0	1	0	0
37	10	240	2	(120, 120)	*	310.5	310.5	310.5	310.5	0.0	0.1	1	0	0
38	10	240	2	(80, 160)	*	310.5	310.5	310.5	310.5	0.0	0.1	1	0	0
39	10	240	3	(80, 80, 80)	*	310.5	310.5	310.5	310.5	0.0	<0.1	1	0	0
40	10	240	3	(40, 80, 120)	*	310.5	310.5	310.5	310.5	0.0	0.1	1	0	0
41	10	240	4	(60, 60, 60, 60)	*	310.5	310.5	310.5	310.5	0.0	<0.1	1	0	0

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
42	10	240	4	(20, 30, 60, 130)	*	310.5	310.5	310.5	310.5	0.0	0.4	1	0	0
43	10	240	2	(120, 120)	*	279.9	279.9	279.9	279.9	0.0	0.1	1	0	0
44	10	240	2	(80, 160)	*	279.9	279.9	279.9	279.9	0.0	0.1	1	0	0
45	10	240	3	(80, 80, 80)	*	279.9	279.9	279.9	279.9	0.0	<0.1	1	0	0
46	10	240	3	(40, 80, 120)	*	279.9	279.9	279.9	279.9	0.0	0.1	1	0	0
47	10	240	4	(60, 60, 60, 60)	*	279.9	279.9	279.9	279.9	0.0	0.1	1	0	0
48	10	240	4	(20, 30, 60, 130)	*	279.9	279.9	279.9	279.9	0.0	0.4	1	0	0
49	10	240	2	(120, 120)	*	269.2	269.2	269.2	269.2	0.0	0.1	1	0	0
50	10	240	2	(80, 160)	*	269.2	269.2	269.2	269.2	0.0	0.1	1	0	0
51	10	240	3	(80, 80, 80)	*	269.2	269.2	269.2	269.2	0.0	<0.1	1	0	0
52	10	240	3	(40, 80, 120)	*	269.2	269.2	269.2	269.2	0.0	0.1	1	0	0
53	10	240	4	(60, 60, 60, 60)	*	269.2	269.2	269.2	269.2	0.0	<0.1	1	0	0
54	10	240	4	(20, 30, 60, 130)	*	269.2	269.2	269.2	269.2	0.0	0.1	1	0	0
55	10	240	2	(120, 120)	*	257.8	257.8	257.8	257.8	0.0	0.1	1	0	0
56	10	240	2	(80, 160)	*	257.8	257.8	257.8	257.8	0.0	0.1	1	0	0
57	10	240	3	(80, 80, 80)	*	257.8	257.8	257.8	257.8	0.0	<0.1	1	0	0
58	10	240	3	(40, 80, 120)	*	257.8	257.8	257.8	257.8	0.0	0.1	1	0	0
59	10	240	4	(60, 60, 60, 60)	*	257.8	257.8	257.8	257.8	0.0	<0.1	1	0	0
60	10	240	4	(20, 30, 60, 130)	*	257.8	257.8	257.8	257.8	0.0	0.3	1	0	0
61	20	120	2	(60, 60)	*	568.5	568.5	564.4	568.5	0.0	0.5	1	300	6
62	20	120	2	(40, 80)	*	568.5	568.5	564.4	568.5	0.0	0.5	1	300	6
63	20	120	3	(40, 40, 40)	*	654.9	654.9	653.1	654.9	0.0	0.5	1	300	2
64	20	120	3	(20, 40, 60)	*	613.6	613.6	609.7	613.6	0.0	0.6	1	300	9
65	20	120	4	(30, 30, 30, 30)	*	602.0	602.0	598.4	602.0	0.0	0.5	1	300	14
66	20	120	4	(10, 15, 30, 65)	*	621.6	621.6	621.2	621.6	0.0	0.9	1	300	10
67	20	120	2	(60, 60)	*	521.4	521.4	517.2	521.4	0.0	0.3	1	37	0
68	20	120	2	(40, 80)	*	521.4	521.4	517.2	521.4	0.0	0.3	1	34	0
69	20	120	3	(40, 40, 40)	*	538.2	538.2	536.2	538.2	0.0	0.5	1	120	0
70	20	120	3	(20, 40, 60)	*	538.2	538.2	528.7	538.2	0.0	1.8	1	300	18
71	20	120	4	(30, 30, 30, 30)	*	547.2	547.2	547.2	547.2	0.0	0.2	1	0	0
72	20	120	4	(10, 15, 30, 65)	*	538.2	538.2	538.2	538.2	0.0	0.9	1	0	0
73	20	120	2	(60, 60)	*	472.6	472.6	457.9	472.6	0.0	3.6	1	300	36
74	20	120	2	(40, 80)	*	472.6	472.6	470.3	472.6	0.0	2.5	1	300	10
75	20	120	3	(40, 40, 40)	*	481.2	481.2	475.1	481.2	0.0	6.1	1	300	10
76	20	120	3	(20, 40, 60)	*	472.6	472.6	470.3	472.6	0.0	16.0	1	300	10
77	20	120	4	(30, 30, 30, 30)	*	472.6	472.6	457.9	472.6	0.0	18.6	1	300	20
78	20	120	4	(10, 15, 30, 65)	*	472.6	472.6	470.3	472.6	0.0	60.8	1	300	10
79	20	120	2	(60, 60)	*	508.4	508.4	484.5	504.1	0.0	121.4	12	300	95
80	20	120	2	(40, 80)	*	508.4	508.4	493.0	504.2	0.0	80.9	7	300	113
81	20	120	3	(40, 40, 40)	*	528.3	528.3	518.2	528.2	0.0	8.6	2	300	77
82	20	120	3	(20, 40, 60)	*	508.4	508.4	493.0	508.4	0.0	2.7	1	160	0
83	20	120	4	(30, 30, 30, 30)	*	509.9	509.9	507.9	509.9	0.0	1.8	1	300	19
84	20	120	4	(10, 15, 30, 65)	*	528.3	528.3	518.2	527.9	0.0	63.0	2	300	75
85	20	120	2	(60, 60)	*	479.3	479.3	471.7	479.3	0.0	0.5	1	300	23
86	20	120	2	(40, 80)	*	479.3	479.3	471.7	479.3	0.0	0.5	1	300	20
87	20	120	3	(40, 40, 40)	*	487.3	487.3	481.1	487.3	0.0	0.6	1	300	9
88	20	120	3	(20, 40, 60)	*	479.3	479.3	473.6	479.3	0.0	0.5	1	61	0
89	20	120	4	(30, 30, 30, 30)	*	482.3	482.3	480.0	482.3	0.0	0.3	1	75	0
90	20	120	4	(10, 15, 30, 65)	*	493.6	493.6	491.2	493.6	0.0	7.5	1	300	22
91	20	240	2	(120, 120)	*	440.8	440.8	437.0	440.8	0.0	0.5	1	45	0
92	20	240	2	(80, 160)	*	440.8	440.8	437.0	440.8	0.0	0.5	1	52	0
93	20	240	3	(80, 80, 80)	*	440.8	440.8	437.0	440.8	0.0	5.2	1	43	0
94	20	240	3	(40, 80, 120)	*	440.8	440.8	437.0	440.8	0.0	7.8	1	43	0
95	20	240	4	(60, 60, 60, 60)	*	440.8	440.8	438.1	440.8	0.0	21.4	1	41	0
96	20	240	4	(20, 30, 60, 130)	*	440.8	440.8	437.9	440.8	0.0	166.6	1	44	0
97	20	240	2	(120, 120)	*	463.6	463.6	450.9	463.6	0.0	1.3	1	300	0
98	20	240	2	(80, 160)	*	463.6	463.6	450.9	463.6	0.0	1.0	1	196	0
99	20	240	3	(80, 80, 80)	*	463.6	463.6	451.1	463.6	0.0	5.9	1	300	10
100	20	240	3	(40, 80, 120)	*	463.6	463.6	450.9	463.6	0.0	8.3	1	156	0
101	20	240	4	(60, 60, 60, 60)	*	463.6	463.6	451.7	463.6	0.0	12.7	1	247	0
102	20	240	4	(20, 30, 60, 130)	*	463.6	463.6	451.1	463.6	0.0	154.9	1	214	0
103	20	240	2	(120, 120)	*	451.6	451.6	418.4	451.6	0.0	17.6	1	300	79
104	20	240	2	(80, 160)	*	451.6	451.6	418.4	451.6	0.0	3.0	1	300	32
105	20	240	3	(80, 80, 80)	*	451.6	451.6	418.4	451.6	0.0	14.1	1	300	9
106	20	240	3	(40, 80, 120)	*	451.6	451.6	418.4	451.6	0.0	33.5	1	300	30
107	20	240	4	(60, 60, 60, 60)	*	451.6	451.6	424.4	451.6	0.0	38.1	1	300	8
108	20	240	4	(20, 30, 60, 130)	*	451.6	451.6	421.3	451.6	0.0	606.0	1	300	16
109	20	240	2	(120, 120)	*	457.4	457.4	446.8	457.4	0.0	0.9	1	104	0
110	20	240	2	(80, 160)	*	457.4	457.4	446.8	457.4	0.0	0.9	1	115	0
111	20	240	3	(80, 80, 80)	*	457.4	457.4	446.8	457.4	0.0	9.2	1	111	0
112	20	240	3	(40, 80, 120)	*	457.4	457.4	446.8	457.4	0.0	13.8	1	111	0
113	20	240	4	(60, 60, 60, 60)	*	457.4	457.4	446.8	457.4	0.0	40.5	1	157	0
114	20	240	4	(20, 30, 60, 130)	*	457.4	457.4	446.8	457.4	0.0	198.7	1	112	0
115	20	240	2	(120, 120)	*	451.0	451.0	425.3	451.0	0.0	1.5	1	122	0
116	20	240	2	(80, 160)	*	451.0	451.0	425.3	451.0	0.0	1.6	1	134	0

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
117	20	240	3	(80, 80, 80)	*	451.0	451.0	425.3	451.0	0.0	40.0	1	177	0
118	20	240	3	(40, 80, 120)	*	451.0	451.0	425.3	451.0	0.0	53.6	1	69	0
119	20	240	4	(60, 60, 60, 60)	*	451.0	451.0	426.5	451.0	0.0	260.5	1	60	0
120	20	240	4	(20, 30, 60, 130)	*	451.0	451.0	426.8	451.0	0.0	1900.2	1	61	0
121	30	120	2	(60, 60)	*	911.8	911.8	874.1	911.8	0.0	2.0	1	300	29
122	30	120	2	(40, 80)	*	911.8	911.8	867.6	911.8	0.0	2.4	1	300	40
123	30	120	3	(40, 40, 40)	*	926.6	926.6	897.5	921.9	0.0	108.6	9	300	141
124	30	120	3	(20, 40, 60)	*	911.8	911.8	877.6	911.8	0.0	5.6	1	300	28
125	30	120	4	(30, 30, 30, 30)	*	929.0	928.9	896.3	926.7	0.0	21.7	2	300	122
126	30	120	4	(10, 15, 30, 65)	*	942.8	942.8	919.9	942.8	0.0	12.8	1	300	63
127	30	120	2	(60, 60)	*	690.5	690.5	668.3	690.5	0.0	1.4	1	300	48
128	30	120	2	(40, 80)	*	690.5	690.5	656.2	690.5	0.0	1.4	1	300	49
129	30	120	3	(40, 40, 40)	*	702.3	702.3	698.9	702.3	0.0	0.8	1	132	0
130	30	120	3	(20, 40, 60)	*	690.5	690.5	672.5	690.5	0.0	2.4	1	300	31
131	30	120	4	(30, 30, 30, 30)	*	702.3	702.3	702.3	702.3	0.0	0.4	1	0	0
132	30	120	4	(10, 15, 30, 65)	*	739.3	739.3	719.1	739.3	0.0	12.2	1	300	69
133	30	120	2	(60, 60)	*	626.2	626.2	618.7	626.2	0.0	0.7	1	300	10
134	30	120	2	(40, 80)	*	624.4	624.4	616.0	624.4	0.0	0.7	1	300	0
135	30	120	3	(40, 40, 40)	*	643.2	643.2	636.6	643.2	0.0	2.8	1	300	30
136	30	120	3	(20, 40, 60)	*	626.2	626.2	618.7	626.2	0.0	1.8	1	263	0
137	30	120	4	(30, 30, 30, 30)	*	626.2	626.2	625.4	626.2	0.0	0.8	1	143	0
138	30	120	4	(10, 15, 30, 65)	*	626.2	626.2	626.2	626.2	0.0	6.1	1	0	0
139	30	120	2	(60, 60)	*	646.1	646.1	634.4	644.1	0.0	16.0	2	300	73
140	30	120	2	(40, 80)	*	643.0	643.0	629.5	643.0	0.0	1.8	1	300	0
141	30	120	3	(40, 40, 40)	*	650.5	650.5	643.3	650.0	0.0	29.4	2	300	37
142	30	120	3	(20, 40, 60)	*	650.5	650.5	639.0	645.3	0.0	141.9	3	300	52
143	30	120	4	(30, 30, 30, 30)	*	650.5	650.5	644.6	650.5	0.0	11.5	1	300	10
144	30	120	4	(10, 15, 30, 65)	*	650.5	650.5	639.7	650.5	0.0	196.0	1	300	50
145	30	120	2	(60, 60)	*	638.7	638.7	628.6	638.7	0.0	5.5	1	300	30
146	30	120	2	(40, 80)	*	638.7	638.7	628.6	638.7	0.0	5.3	1	300	30
147	30	120	3	(40, 40, 40)	*	656.9	656.9	640.0	652.2	0.0	741.3	4	300	246
148	30	120	3	(20, 40, 60)	*	638.7	638.7	628.6	638.7	0.0	45.4	1	300	20
149	30	120	4	(30, 30, 30, 30)	*	638.7	638.7	636.3	638.7	0.0	29.6	1	300	10
150	30	120	4	(10, 15, 30, 65)		656.9	655.6	640.0	652.6	0.2	*TL*	2	300	215
151	30	240	2	(120, 120)	*	582.7	582.7	543.3	582.7	0.0	13.4	1	300	25
152	30	240	2	(80, 160)	*	582.7	582.7	543.3	582.7	0.0	8.7	1	300	10
153	30	240	3	(80, 80, 80)	*	582.7	582.7	543.4	582.7	0.0	99.3	1	300	10
154	30	240	3	(40, 80, 120)	*	582.7	582.7	543.3	582.7	0.0	212.8	1	300	26
155	30	240	4	(60, 60, 60, 60)	*	582.7	582.7	543.7	582.7	0.0	785.7	1	300	23
156	30	240	4	(20, 30, 60, 130)	*	582.7	582.7	543.4	582.7	0.0	1727.4	1	300	16
157	30	240	2	(120, 120)	*	511.1	511.1	491.1	511.1	0.0	366.1	1	300	53
158	30	240	2	(80, 160)	*	511.1	511.1	491.1	511.1	0.0	314.2	1	300	30
159	30	240	3	(80, 80, 80)	*	511.1	511.1	491.1	511.1	0.0	2599.1	1	300	30
160	30	240	3	(40, 80, 120)			501.3	491.1	501.3		*TL*	1	300	30
161	30	240	4	(60, 60, 60, 60)			491.2	491.2	491.2		*TL*	1	113	0
162	30	240	4	(20, 30, 60, 130)							*TL*	0	0	0
163	30	240	2	(120, 120)	*	467.7	467.7	458.8	467.7	0.0	2.9	1	198	0
164	30	240	2	(80, 160)	*	467.7	467.7	458.8	467.7	0.0	4.8	1	191	0
165	30	240	3	(80, 80, 80)	*	467.7	467.7	458.8	467.7	0.0	73.4	1	203	0
166	30	240	3	(40, 80, 120)	*	467.7	467.7	458.8	467.7	0.0	112.4	1	185	0
167	30	240	4	(60, 60, 60, 60)	*	467.7	467.7	458.8	467.7	0.0	760.6	1	181	0
168	30	240	4	(20, 30, 60, 130)	*	467.7	467.7	458.8	467.7	0.0	2061.3	1	204	0
169	30	240	2	(120, 120)	*	541.7	541.7	513.2	541.7	0.0	38.0	1	300	10
170	30	240	2	(80, 160)	*	541.7	541.7	513.2	541.7	0.0	35.5	1	275	0
171	30	240	3	(80, 80, 80)	*	541.7	541.7	513.2	541.7	0.0	835.7	1	300	20
172	30	240	3	(40, 80, 120)	*	541.7	541.7	513.2	541.7	0.0	1305.1	1	300	10
173	30	240	4	(60, 60, 60, 60)			513.2	513.2	513.2		*TL*	1	0	0
174	30	240	4	(20, 30, 60, 130)							*TL*	0	0	0
175	30	240	2	(120, 120)	*	623.4	623.4	593.7	623.4	0.0	15.6	1	300	50
176	30	240	2	(80, 160)	*	623.4	623.4	593.7	623.4	0.0	18.0	1	300	57
177	30	240	3	(80, 80, 80)	*	623.4	623.4	593.8	623.4	0.0	235.3	1	300	50
178	30	240	3	(40, 80, 120)	*	623.4	623.4	593.7	623.4	0.0	319.9	1	300	51
179	30	240	4	(60, 60, 60, 60)	*	630.3	630.3	594.8	630.3	0.0	3400.3	1	300	155
180	30	240	4	(20, 30, 60, 130)			619.9	594.7	619.9		*TL*	1	300	30
181	40	120	2	(60, 60)	*	761.7	761.7	738.5	761.7	0.0	2.1	1	300	30
182	40	120	2	(40, 80)	*	761.7	761.7	738.5	761.7	0.0	2.3	1	300	31
183	40	120	3	(40, 40, 40)	*	782.0	781.9	748.8	778.8	0.0	242.0	6	300	186
184	40	120	3	(20, 40, 60)	*	773.0	773.0	740.9	773.0	0.0	21.4	1	300	73
185	40	120	4	(30, 30, 30, 30)	*	782.7	782.7	762.4	782.7	0.0	32.1	1	300	124
186	40	120	4	(10, 15, 30, 65)	*	782.7	782.7	765.9	782.7	0.0	88.0	1	300	76
187	40	120	2	(60, 60)	*	847.9	847.9	832.5	847.9	0.0	8.1	1	300	82
188	40	120	2	(40, 80)	*	847.9	847.9	826.8	846.8	0.0	24.1	2	300	115
189	40	120	3	(40, 40, 40)	*	874.6	874.6	864.2	868.5	0.0	111.9	5	300	121
190	40	120	3	(20, 40, 60)	*	874.0	873.9	860.6	865.3	0.0	916.2	16	300	226
191	40	120	4	(30, 30, 30, 30)	*	878.8	878.8	851.0	869.5	0.0	3070.7	34	300	297

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
192	40	120	4	(10, 15, 30, 65)	*	887.6	887.6	874.0	882.7	0.0	467.4	4	300	131
193	40	120	2	(60, 60)	*	749.0	749.0	723.5	749.0	0.0	5.8	1	300	106
194	40	120	2	(40, 80)	*	748.2	748.2	722.5	748.2	0.0	12.2	2	300	82
195	40	120	3	(40, 40, 40)	*	767.0	767.0	741.5	767.0	0.0	16.0	1	300	101
196	40	120	3	(20, 40, 60)	*	749.0	749.0	723.5	749.0	0.0	19.0	1	300	70
197	40	120	4	(30, 30, 30, 30)	*	790.0	789.9	773.6	787.6	0.0	37.3	3	300	97
198	40	120	4	(10, 15, 30, 65)	*	787.2	787.2	778.1	787.2	0.0	20.0	1	300	20
199	40	120	2	(60, 60)	*	911.4	911.4	898.3	911.4	0.0	1.5	1	300	10
200	40	120	2	(40, 80)	*	911.4	911.4	898.3	911.4	0.0	2.3	1	300	35
201	40	120	3	(40, 40, 40)	*	984.6	984.6	969.3	978.5	0.0	26.6	2	300	93
202	40	120	3	(20, 40, 60)	*	944.0	944.0	927.9	944.0	0.0	28.0	1	300	33
203	40	120	4	(30, 30, 30, 30)	*	958.2	958.2	940.5	956.0	0.0	52.2	2	300	140
204	40	120	4	(10, 15, 30, 65)			991.1	979.7	991.1		*TL*	2	300	155
205	40	120	2	(60, 60)	*	909.6	909.6	883.5	909.3	0.0	119.5	3	300	155
206	40	120	2	(40, 80)	*	909.6	909.6	883.4	908.6	0.0	110.0	3	300	149
207	40	120	3	(40, 40, 40)	*	917.2	917.2	893.9	917.2	0.0	103.9	1	300	88
208	40	120	3	(20, 40, 60)	*	909.6	909.5	884.6	909.5	0.0	428.6	2	300	165
209	40	120	4	(30, 30, 30, 30)	*	931.8	931.8	909.9	929.7	0.0	480.5	5	300	147
210	40	120	4	(10, 15, 30, 65)	*	922.4	922.4	895.5	921.3	0.0	1696.8	4	300	138
211	40	240	2	(120, 120)	*	615.5	615.5	605.1	615.5	0.0	21.4	1	300	30
212	40	240	2	(80, 160)	*	615.5	615.5	605.1	615.5	0.0	32.5	1	300	52
213	40	240	3	(80, 80, 80)	*	615.5	615.5	605.3	615.5	0.0	326.8	1	300	43
214	40	240	3	(40, 80, 120)	*	615.5	615.5	605.1	615.5	0.0	1530.1	1	300	66
215	40	240	4	(60, 60, 60, 60)	*	615.5	615.5	606.2	615.5	0.0	875.9	1	300	40
216	40	240	4	(20, 30, 60, 130)							*TL*	0	0	0
217	40	240	2	(120, 120)	*	676.7	676.7	671.4	676.7	0.0	23.7	1	300	31
218	40	240	2	(80, 160)	*	676.7	676.7	671.4	676.7	0.0	31.4	1	300	30
219	40	240	3	(80, 80, 80)	*	676.7	676.7	671.5	676.7	0.0	195.3	1	300	29
220	40	240	3	(40, 80, 120)	*	676.7	676.7	671.4	676.7	0.0	385.1	1	300	30
221	40	240	4	(60, 60, 60, 60)			726.9	714.6	726.9		*TL*	1	300	126
222	40	240	4	(20, 30, 60, 130)			715.4	710.0	715.4		*TL*	1	300	10
223	40	240	2	(120, 120)			662.7	647.1	662.7		*TL*	1	300	215
224	40	240	2	(80, 160)			667.0	647.1	667.0		*TL*	1	300	205
225	40	240	3	(80, 80, 80)			662.2	647.1	662.2		*TL*	1	300	60
226	40	240	3	(40, 80, 120)			656.1	647.1	656.1		*TL*	1	300	40
227	40	240	4	(60, 60, 60, 60)			648.1	648.1	648.1		*TL*	1	300	0
228	40	240	4	(20, 30, 60, 130)							*TL*	0	0	0
229	40	240	2	(120, 120)	*	659.5	659.5	647.5	659.5	0.0	43.0	1	300	50
230	40	240	2	(80, 160)	*	659.5	659.5	647.5	659.5	0.0	52.7	1	300	64
231	40	240	3	(80, 80, 80)	*	659.5	659.5	648.8	659.5	0.0	572.3	1	300	61
232	40	240	3	(40, 80, 120)	*	659.5	659.5	647.5	659.5	0.0	594.6	1	300	42
233	40	240	4	(60, 60, 60, 60)	*	659.5	659.5	649.5	659.5	0.0	1204.0	1	300	30
234	40	240	4	(20, 30, 60, 130)			652.8	649.3	652.8		*TL*	1	300	0
235	40	240	2	(120, 120)	*	700.4	700.4	689.9	700.4	0.0	95.5	1	300	83
236	40	240	2	(80, 160)	*	700.4	700.4	689.9	700.4	0.0	110.7	1	300	71
237	40	240	3	(80, 80, 80)	*	700.4	700.4	690.4	700.4	0.0	1689.8	1	300	70
238	40	240	3	(40, 80, 120)	*	700.4	700.4	689.9	700.4	0.0	2654.5	1	300	72
239	40	240	4	(60, 60, 60, 60)			699.3	690.9	699.3		*TL*	1	300	32
240	40	240	4	(20, 30, 60, 130)							*TL*	0	0	0
241	50	120	2	(60, 60)		1345.8	1268.9	1244.4	1263.7	6.0	*TL*	25	300	320
242	50	120	2	(40, 80)		1342.3	1260.3	1236.4	1252.9	6.5	*TL*	15	300	320
243	50	120	3	(40, 40, 40)		1365.3	1293.5	1263.3	1289.5	5.5	*TL*	13	300	320
244	50	120	3	(20, 40, 60)		1333.4	1315.0	1257.8	1311.8	1.4	*TL*	4	300	303
245	50	120	4	(30, 30, 30, 30)		1358.6	1353.2	1339.1	1345.6	0.4	*TL*	26	300	320
246	50	120	4	(10, 15, 30, 65)		1380.7	1309.4	1287.2	1308.0	5.4	*TL*	3	300	198
247	50	120	2	(60, 60)	*	978.4	978.4	959.2	978.1	0.0	31.1	2	300	150
248	50	120	2	(40, 80)	*	978.4	978.4	957.9	976.5	0.0	87.3	3	300	150
249	50	120	3	(40, 40, 40)	*	1066.8	1066.7	1042.1	1061.7	0.0	1228.5	15	300	270
250	50	120	3	(20, 40, 60)		1020.3	1015.2	992.0	1007.5	0.5	*TL*	37	300	320
251	50	120	4	(30, 30, 30, 30)	*	1021.8	1021.8	1003.5	1021.8	0.0	55.9	1	300	79
252	50	120	4	(10, 15, 30, 65)		1055.8	1053.6	1037.0	1052.8	0.2	*TL*	3	300	146
253	50	120	2	(60, 60)	*	974.0	974.0	945.9	967.9	0.0	3333.7	22	300	320
254	50	120	2	(40, 80)		974.0	971.5	941.6	963.7	0.3	*TL*	22	300	320
255	50	120	3	(40, 40, 40)	*	1024.7	1024.7	997.0	1017.8	0.0	223.2	2	300	144
256	50	120	3	(20, 40, 60)	*	989.8	989.8	968.9	989.8	0.0	337.0	2	300	132
257	50	120	4	(30, 30, 30, 30)		1018.7	1016.3	981.5	1001.8	0.2	*TL*	13	300	277
258	50	120	4	(10, 15, 30, 65)	*	991.1	991.1	973.3	990.9	0.0	2114.0	3	300	165
259	50	120	2	(60, 60)	*	1035.3	1035.3	1011.9	1035.3	0.0	30.0	1	300	100
260	50	120	2	(40, 80)	*	1030.1	1030.1	1011.7	1030.1	0.0	29.0	1	300	72
261	50	120	3	(40, 40, 40)	*	1079.9	1079.9	1071.8	1079.9	0.0	57.4	1	300	30
262	50	120	3	(20, 40, 60)	*	1051.5	1051.4	1024.9	1043.8	0.0	1463.7	6	300	229
263	50	120	4	(30, 30, 30, 30)		1079.7	1077.1	1044.8	1067.7	0.2	*TL*	9	300	320
264	50	120	4	(10, 15, 30, 65)			1083.9	1062.8	1083.9		*TL*	1	300	110
265	50	120	2	(60, 60)		990.9	970.4	940.4	964.6	2.1	*TL*	18	300	320
266	50	120	2	(40, 80)		990.9	969.1	939.2	963.1	2.2	*TL*	13	300	320

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
267	50	120	3	(40, 40, 40)		1012.7	1007.4	980.2	997.0	0.5	*TL*	40	300	320
268	50	120	3	(20, 40, 60)		996.9	975.7	947.0	972.6	2.1	*TL*	7	300	320
269	50	120	4	(30, 30, 30, 30)		1077.0	1071.1	1055.6	1061.9	0.5	*TL*	56	300	320
270	50	120	4	(10, 15, 30, 65)		1084.5	1079.2	1065.0	1075.8	0.5	*TL*	3	300	147
271	50	240	2	(120, 120)	*	663.3	663.3	658.2	663.3	0.0	17.5	1	300	20
272	50	240	2	(80, 160)	*	663.3	663.3	658.2	663.3	0.0	19.6	1	300	20
273	50	240	3	(80, 80, 80)	*	663.3	663.3	658.2	663.3	0.0	401.8	1	300	30
274	50	240	3	(40, 80, 120)	*	663.3	663.3	658.2	663.3	0.0	582.8	1	300	20
275	50	240	4	(60, 60, 60, 60)			663.3	658.9	663.3		*TL*	1	300	20
276	50	240	4	(20, 30, 60, 130)							*TL*	0	0	0
277	50	240	2	(120, 120)	*	883.6	883.6	869.7	883.6	0.0	54.6	1	300	40
278	50	240	2	(80, 160)	*	883.6	883.6	869.7	883.6	0.0	41.9	1	300	30
279	50	240	3	(80, 80, 80)	*	890.5	890.5	871.8	890.5	0.0	1644.4	1	300	114
280	50	240	3	(40, 80, 120)	*	883.6	883.6	870.0	883.6	0.0	670.5	1	300	30
281	50	240	4	(60, 60, 60, 60)			895.1	877.1	895.1		*TL*	1	300	90
282	50	240	4	(20, 30, 60, 130)			883.4	872.7	883.4		*TL*	1	300	30
283	50	240	2	(120, 120)	*	875.1	875.1	861.1	875.1	0.0	467.8	1	300	132
284	50	240	2	(80, 160)	*	875.1	875.1	861.1	875.1	0.0	435.9	1	300	122
285	50	240	3	(80, 80, 80)			873.9	861.5	873.9		*TL*	1	300	101
286	50	240	3	(40, 80, 120)			872.3	861.4	872.3		*TL*	1	300	82
287	50	240	4	(60, 60, 60, 60)			877.2	868.7	877.2		*TL*	1	300	10
288	50	240	4	(20, 30, 60, 130)			861.5	861.5	861.5		*TL*	1	300	0
289	50	240	2	(120, 120)		1119.5	1026.3	993.8	1023.3	9.0	*TL*	4	300	320
290	50	240	2	(80, 160)		1112.4	1010.8	993.8	1008.1	10.0	*TL*	3	300	275
291	50	240	3	(80, 80, 80)			1011.6	997.2	1011.6		*TL*	1	300	102
292	50	240	3	(40, 80, 120)			1010.6	994.2	1010.6		*TL*	1	300	50
293	50	240	4	(60, 60, 60, 60)			1021.4	1011.6	1021.4		*TL*	1	300	10
294	50	240	4	(20, 30, 60, 130)			998.7	998.7	998.7		*TL*	1	0	0
295	50	240	2	(120, 120)	*	633.0	633.0	625.5	633.0	0.0	144.7	1	300	20
296	50	240	2	(80, 160)	*	633.0	633.0	625.5	633.0	0.0	270.4	1	300	40
297	50	240	3	(80, 80, 80)			632.6	625.5	632.6		*TL*	1	300	30
298	50	240	3	(40, 80, 120)			627.9	625.5	627.9		*TL*	1	300	0
299	50	240	4	(60, 60, 60, 60)							*TL*	0	0	0
300	50	240	4	(20, 30, 60, 130)							*TL*	0	0	0
301	60	120	2	(60, 60)	*	989.1	989.1	967.7	980.4	0.0	3272.7	19	300	320
302	60	120	2	(40, 80)		989.1	988.4	967.0	980.3	0.1	*TL*	20	300	320
303	60	120	3	(40, 40, 40)		1008.7	1004.3	980.4	997.6	0.4	*TL*	6	300	320
304	60	120	3	(20, 40, 60)		993.4	985.2	969.3	982.2	0.8	*TL*	3	300	229
305	60	120	4	(30, 30, 30, 30)		1016.0	1010.4	990.2	1007.5	0.5	*TL*	3	300	252
306	60	120	4	(10, 15, 30, 65)			998.0	984.9	998.0		*TL*	1	300	126
307	60	120	2	(60, 60)	*	1114.8	1114.8	1094.7	1114.2	0.0	65.6	2	300	190
308	60	120	2	(40, 80)	*	1121.6	1121.6	1097.7	1119.2	0.0	470.0	8	300	264
309	60	120	3	(40, 40, 40)		1167.9	1163.3	1130.4	1157.5	0.4	*TL*	35	300	320
310	60	120	3	(20, 40, 60)	*	1141.7	1141.7	1115.7	1139.0	0.0	860.2	6	300	298
311	60	120	4	(30, 30, 30, 30)	*	1167.6	1167.6	1147.3	1166.8	0.0	625.5	3	300	178
312	60	120	4	(10, 15, 30, 65)		1169.0	1164.3	1135.0	1160.5	0.4	*TL*	16	300	277
313	60	120	2	(60, 60)	*	957.1	957.1	932.3	954.5	0.0	1727.5	9	300	320
314	60	120	2	(40, 80)	*	954.6	954.5	931.8	952.8	0.0	848.7	5	300	307
315	60	120	3	(40, 40, 40)		965.8	965.4	937.3	960.0	0.0	*TL*	10	300	320
316	60	120	3	(20, 40, 60)			957.1	933.8	957.1		*TL*	3	300	273
317	60	120	4	(30, 30, 30, 30)	*	963.1	963.1	949.4	963.1	0.0	525.8	1	300	85
318	60	120	4	(10, 15, 30, 65)			959.8	943.6	959.8		*TL*	1	300	93
319	60	120	2	(60, 60)	*	1002.5	1002.5	979.2	998.9	0.0	1048.2	5	300	320
320	60	120	2	(40, 80)	*	1002.5	1002.5	979.0	998.3	0.0	2776.8	15	300	320
321	60	120	3	(40, 40, 40)		1047.4	1032.8	1004.5	1028.2	1.4	*TL*	7	300	320
322	60	120	3	(20, 40, 60)		1005.9	1000.9	982.2	1000.6	0.5	*TL*	2	300	257
323	60	120	4	(30, 30, 30, 30)		1049.9	1041.8	1016.3	1039.6	0.8	*TL*	6	300	320
324	60	120	4	(10, 15, 30, 65)			1021.2	1009.9	1021.2		*TL*	1	300	75
325	60	120	2	(60, 60)	*	1063.9	1063.9	1038.1	1058.7	0.0	1583.8	13	300	320
326	60	120	2	(40, 80)		1063.9	1063.0	1037.4	1057.2	0.1	*TL*	31	300	320
327	60	120	3	(40, 40, 40)		1069.2	1064.2	1042.7	1062.3	0.5	*TL*	4	300	303
328	60	120	3	(20, 40, 60)		1063.9	1062.4	1039.4	1059.9	0.1	*TL*	4	300	279
329	60	120	4	(30, 30, 30, 30)			1088.1	1076.2	1088.1		*TL*	2	300	195
330	60	120	4	(10, 15, 30, 65)			1067.4	1060.2	1067.4		*TL*	1	300	42
331	60	240	2	(120, 120)		943.3	928.3	915.1	927.4	1.6	*TL*	2	300	203
332	60	240	2	(80, 160)		934.1	929.9	915.1	929.8	0.5	*TL*	3	300	224
333	60	240	3	(80, 80, 80)			923.4	915.2	923.4		*TL*	1	300	40
334	60	240	3	(40, 80, 120)			921.0	915.1	921.0		*TL*	1	300	20
335	60	240	4	(60, 60, 60, 60)			918.3	918.3	918.3		*TL*	1	0	0
336	60	240	4	(20, 30, 60, 130)							*TL*	0	0	0
337	60	240	2	(120, 120)	*	915.6	915.6	898.0	915.6	0.0	2046.5	1	300	239
338	60	240	2	(80, 160)	*	915.6	915.6	898.0	915.6	0.0	1845.7	1	300	218
339	60	240	3	(80, 80, 80)			912.0	898.9	912.0		*TL*	1	300	80
340	60	240	3	(40, 80, 120)			909.1	898.8	909.1		*TL*	1	300	50
341	60	240	4	(60, 60, 60, 60)			905.7	903.1	905.7		*TL*	1	300	10

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
342	60	240	4	(20, 30, 60, 130)							*TL*	0	0	0
343	60	240	2	(120, 120)		823.5	815.5	791.1	812.8	1.0	*TL*	10	300	320
344	60	240	2	(80, 160)		820.6	815.0	791.1	812.9	0.7	*TL*	8	300	320
345	60	240	3	(80, 80, 80)			816.0	791.9	816.0		*TL*	1	300	170
346	60	240	3	(40, 80, 120)			815.4	791.4	815.4		*TL*	1	300	148
347	60	240	4	(60, 60, 60, 60)			803.4	802.1	803.4		*TL*	1	300	20
348	60	240	4	(20, 30, 60, 130)							*TL*	0	0	0
349	60	240	2	(120, 120)			1117.1	1106.5	1117.1		*TL*	2	300	239
350	60	240	2	(80, 160)			1116.7	1106.5	1116.7		*TL*	1	300	243
351	60	240	3	(80, 80, 80)			1112.6	1108.8	1112.6		*TL*	1	300	20
352	60	240	3	(40, 80, 120)			1110.5	1106.9	1110.5		*TL*	1	300	23
353	60	240	4	(60, 60, 60, 60)							*TL*	0	0	0
354	60	240	4	(20, 30, 60, 130)							*TL*	0	0	0
355	60	240	2	(120, 120)	*	764.3	764.3	748.6	764.3	0.0	1109.3	1	300	134
356	60	240	2	(80, 160)	*	764.3	764.3	748.6	764.3	0.0	1128.9	1	300	127
357	60	240	3	(80, 80, 80)			755.8	748.6	755.8		*TL*	1	300	30
358	60	240	3	(40, 80, 120)			754.0	748.6	754.0		*TL*	1	300	20
359	60	240	4	(60, 60, 60, 60)							*TL*	0	0	0
360	60	240	4	(20, 30, 60, 130)							*TL*	0	0	0
361	70	120	2	(60, 60)		1347.2	1339.5	1313.1	1334.0	0.6	*TL*	17	300	320
362	70	120	2	(40, 80)		1334.1	1333.7	1309.8	1330.1	0.0	*TL*	14	300	320
363	70	120	3	(40, 40, 40)	*	1367.3	1367.3	1347.9	1367.3	0.0	1592.0	1	300	194
364	70	120	3	(20, 40, 60)			1340.4	1324.4	1340.4		*TL*	3	300	164
365	70	120	4	(30, 30, 30, 30)		1384.7	1381.4	1366.5	1377.8	0.2	*TL*	2	300	212
366	70	120	4	(10, 15, 30, 65)			1383.3	1365.3	1383.3		*TL*	1	300	121
367	70	120	2	(60, 60)		1292.2	1272.7	1246.6	1262.7	1.5	*TL*	16	300	320
368	70	120	2	(40, 80)		1278.7	1267.8	1243.6	1258.6	0.9	*TL*	14	300	320
369	70	120	3	(40, 40, 40)			1298.2	1285.7	1298.2		*TL*	2	300	167
370	70	120	3	(20, 40, 60)			1265.9	1248.5	1265.9		*TL*	2	300	221
371	70	120	4	(30, 30, 30, 30)	*	1329.2	1329.2	1315.0	1329.2	0.0	323.9	2	300	183
372	70	120	4	(10, 15, 30, 65)			1316.5	1302.6	1316.5		*TL*	2	300	184
373	70	120	2	(60, 60)		1351.0	1343.6	1324.0	1342.6	0.5	*TL*	2	300	319
374	70	120	2	(40, 80)		1357.2	1342.2	1324.6	1341.4	1.1	*TL*	3	300	320
375	70	120	3	(40, 40, 40)			1383.4	1371.6	1383.4		*TL*	1	300	159
376	70	120	3	(20, 40, 60)			1363.3	1352.9	1363.3		*TL*	1	300	93
377	70	120	4	(30, 30, 30, 30)			1388.7	1380.2	1388.7		*TL*	1	300	36
378	70	120	4	(10, 15, 30, 65)			1380.7	1380.7	1380.7		*TL*	1	300	0
379	70	120	2	(60, 60)	*	1109.5	1109.5	1095.8	1106.8	0.0	808.5	5	300	294
380	70	120	2	(40, 80)	*	1109.5	1109.4	1088.5	1104.3	0.0	2390.7	24	300	320
381	70	120	3	(40, 40, 40)		1162.2	1159.3	1139.3	1153.8	0.2	*TL*	23	300	320
382	70	120	3	(20, 40, 60)	*	1131.9	1131.9	1113.8	1129.2	0.0	3135.4	5	300	278
383	70	120	4	(30, 30, 30, 30)		1214.3	1209.6	1186.3	1202.6	0.4	*TL*	10	300	320
384	70	120	4	(10, 15, 30, 65)		1193.8	1182.9	1160.6	1181.2	0.9	*TL*	2	300	250
385	70	120	2	(60, 60)	*	1073.7	1073.6	1047.4	1071.6	0.0	1817.8	6	300	320
386	70	120	2	(40, 80)	*	1072.9	1072.9	1047.4	1069.8	0.0	2790.6	14	300	320
387	70	120	3	(40, 40, 40)	*	1080.7	1080.7	1058.8	1076.2	0.0	2732.7	4	300	248
388	70	120	3	(20, 40, 60)			1074.7	1049.4	1074.7		*TL*	1	300	241
389	70	120	4	(30, 30, 30, 30)			1094.3	1075.3	1094.3		*TL*	2	300	215
390	70	120	4	(10, 15, 30, 65)			1088.6	1067.4	1088.6		*TL*	1	300	148
391	70	240	2	(120, 120)	*	970.1	970.1	958.6	970.1	0.0	427.4	1	300	72
392	70	240	2	(80, 160)	*	970.1	970.1	958.6	970.1	0.0	862.0	1	300	112
393	70	240	3	(80, 80, 80)			965.0	958.6	965.0		*TL*	1	300	10
394	70	240	3	(40, 80, 120)			964.5	958.6	964.5		*TL*	1	300	10
395	70	240	4	(60, 60, 60, 60)							*TL*	0	0	0
396	70	240	4	(20, 30, 60, 130)							*TL*	0	0	0
397	70	240	2	(120, 120)	*	1009.6	1009.6	985.7	1007.4	0.0	3595.9	3	300	305
398	70	240	2	(80, 160)		1009.6	1009.3	985.7	1007.6	0.0	*TL*	3	300	300
399	70	240	3	(80, 80, 80)			1005.0	990.2	1005.0		*TL*	1	300	94
400	70	240	3	(40, 80, 120)			1002.3	986.5	1002.3		*TL*	1	300	77
401	70	240	4	(60, 60, 60, 60)			1003.0	1001.0	1003.0		*TL*	1	300	10
402	70	240	4	(20, 30, 60, 130)			818.6	818.6	818.6		*TL*	1	0	0
403	70	240	2	(120, 120)			1131.7	1109.4	1131.7		*TL*	1	300	204
404	70	240	2	(80, 160)			1131.4	1109.4	1131.4		*TL*	1	300	186
405	70	240	3	(80, 80, 80)			1118.1	1109.5	1118.1		*TL*	1	300	10
406	70	240	3	(40, 80, 120)			1112.4	1109.4	1112.4		*TL*	1	300	0
407	70	240	4	(60, 60, 60, 60)							*TL*	0	0	0
408	70	240	4	(20, 30, 60, 130)							*TL*	0	0	0
409	70	240	2	(120, 120)			1001.5	987.1	1001.5		*TL*	1	300	171
410	70	240	2	(80, 160)			1001.0	987.1	1001.0		*TL*	1	300	171
411	70	240	3	(80, 80, 80)			990.8	987.6	990.8		*TL*	1	300	10
412	70	240	3	(40, 80, 120)			720.2	720.2	720.2		*TL*	1	0	0
413	70	240	4	(60, 60, 60, 60)							*TL*	0	0	0
414	70	240	4	(20, 30, 60, 130)							*TL*	0	0	0
415	70	240	2	(120, 120)			958.5	937.2	958.5		*TL*	1	300	259
416	70	240	2	(80, 160)			958.4	937.2	958.4		*TL*	1	300	257

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
417	70	240	3	(80, 80, 80)			945.8	937.2	945.8		*TL*	1	300	20
418	70	240	3	(40, 80, 120)			945.8	937.2	945.8		*TL*	1	300	10
419	70	240	4	(60, 60, 60, 60)			954.9	954.9	954.9		*TL*	1	0	0
420	70	240	4	(20, 30, 60, 130)							*TL*	0	0	0
421	80	120	2	(60, 60)		1414.5	1410.1	1385.1	1405.7	0.3	*TL*	24	300	320
422	80	120	2	(40, 80)		1413.4	1409.0	1384.0	1405.1	0.3	*TL*	23	300	320
423	80	120	3	(40, 40, 40)		1428.2	1423.8	1404.2	1420.8	0.3	*TL*	13	300	320
424	80	120	3	(20, 40, 60)		1414.7	1412.5	1389.9	1411.4	0.2	*TL*	5	300	279
425	80	120	4	(30, 30, 30, 30)		1499.7	1489.6	1479.4	1486.6	0.7	*TL*	6	300	320
426	80	120	4	(10, 15, 30, 65)		1462.2	1455.5	1440.5	1455.2	0.5	*TL*	3	300	223
427	80	120	2	(60, 60)		1325.3	1322.7	1299.2	1317.8	0.2	*TL*	22	300	320
428	80	120	2	(40, 80)		1330.6	1322.5	1300.9	1319.6	0.6	*TL*	9	300	320
429	80	120	3	(40, 40, 40)	*	1373.4	1373.4	1361.9	1373.4	0.0	295.8	1	300	106
430	80	120	3	(20, 40, 60)			1334.5	1320.7	1334.5		*TL*	2	300	263
431	80	120	4	(30, 30, 30, 30)	*	1410.7	1410.7	1389.2	1410.3	0.0	1936.2	3	300	295
432	80	120	4	(10, 15, 30, 65)	*	1386.1	1386.1	1366.4	1386.1	0.0	1775.3	1	300	108
433	80	120	2	(60, 60)		1570.0	1525.1	1505.4	1523.7	2.9	*TL*	4	300	262
434	80	120	2	(40, 80)		1557.7	1525.4	1505.0	1524.2	2.1	*TL*	5	300	248
435	80	120	3	(40, 40, 40)		1585.7	1566.8	1547.8	1564.6	1.2	*TL*	6	300	320
436	80	120	3	(20, 40, 60)		1570.5	1532.1	1515.5	1531.1	2.5	*TL*	3	300	218
437	80	120	4	(30, 30, 30, 30)		1576.5	1569.6	1558.6	1566.1	0.4	*TL*	5	300	261
438	80	120	4	(10, 15, 30, 65)			1585.2	1572.0	1585.2		*TL*	1	300	193
439	80	120	2	(60, 60)		1575.6	1572.1	1554.2	1567.6	0.2	*TL*	19	300	320
440	80	120	2	(40, 80)	*	1561.0	1561.0	1545.2	1558.0	0.0	1771.6	9	300	304
441	80	120	3	(40, 40, 40)		1617.1	1611.2	1590.6	1607.8	0.4	*TL*	2	300	297
442	80	120	3	(20, 40, 60)			1573.3	1557.0	1573.3		*TL*	1	300	229
443	80	120	4	(30, 30, 30, 30)			1647.4	1627.7	1647.4		*TL*	1	300	202
444	80	120	4	(10, 15, 30, 65)			1621.2	1608.9	1621.2		*TL*	1	300	81
445	80	120	2	(60, 60)		1447.6	1436.6	1414.1	1429.8	0.8	*TL*	12	300	320
446	80	120	2	(40, 80)		1447.6	1434.7	1412.3	1428.1	0.9	*TL*	9	300	320
447	80	120	3	(40, 40, 40)			1448.9	1430.3	1448.9		*TL*	1	300	211
448	80	120	3	(20, 40, 60)			1430.3	1415.2	1430.3		*TL*	1	300	182
449	80	120	4	(30, 30, 30, 30)			1495.5	1479.0	1495.5		*TL*	1	300	136
450	80	120	4	(10, 15, 30, 65)			1464.7	1454.0	1464.7		*TL*	1	300	56
451	80	240	2	(120, 120)			1340.6	1326.6	1340.6		*TL*	1	300	209
452	80	240	2	(80, 160)			1340.7	1326.6	1340.7		*TL*	1	300	207
453	80	240	3	(80, 80, 80)			1333.3	1329.7	1333.3		*TL*	1	300	10
454	80	240	3	(40, 80, 120)			1328.5	1328.5	1328.5		*TL*	1	300	0
455	80	240	4	(60, 60, 60, 60)							*TL*	0	0	0
456	80	240	4	(20, 30, 60, 130)							*TL*	0	0	0
457	80	240	2	(120, 120)			1159.5	1138.8	1159.5		*TL*	1	300	191
458	80	240	2	(80, 160)			1156.4	1138.7	1156.4		*TL*	1	300	240
459	80	240	3	(80, 80, 80)			1145.1	1140.4	1145.1		*TL*	1	300	20
460	80	240	3	(40, 80, 120)			1139.0	1139.0	1139.0		*TL*	1	300	0
461	80	240	4	(60, 60, 60, 60)			1152.5	1152.5	1152.5		*TL*	1	0	0
462	80	240	4	(20, 30, 60, 130)							*TL*	0	0	0
463	80	240	2	(120, 120)			1328.6	1313.9	1328.6		*TL*	1	300	286
464	80	240	2	(80, 160)			1327.8	1313.9	1327.8		*TL*	1	300	262
465	80	240	3	(80, 80, 80)			1323.2	1314.6	1323.2		*TL*	1	300	30
466	80	240	3	(40, 80, 120)			1320.7	1314.1	1320.7		*TL*	1	300	10
467	80	240	4	(60, 60, 60, 60)			1322.9	1322.9	1322.9		*TL*	1	0	0
468	80	240	4	(20, 30, 60, 130)							*TL*	0	0	0
469	80	240	2	(120, 120)			1168.0	1156.7	1168.0		*TL*	1	300	169
470	80	240	2	(80, 160)			1168.2	1156.7	1168.2		*TL*	1	300	165
471	80	240	3	(80, 80, 80)			1162.1	1157.2	1162.1		*TL*	1	300	30
472	80	240	3	(40, 80, 120)			1158.8	1157.5	1158.8		*TL*	1	300	20
473	80	240	4	(60, 60, 60, 60)			755.7	755.7	755.7		*TL*	1	0	0
474	80	240	4	(20, 30, 60, 130)							*TL*	0	0	0
475	80	240	2	(120, 120)		1323.6	1315.5	1298.3	1313.0	0.6	*TL*	4	300	320
476	80	240	2	(80, 160)		1326.9	1315.1	1298.3	1313.1	0.9	*TL*	3	300	286
477	80	240	3	(80, 80, 80)			1316.0	1305.1	1316.0		*TL*	1	300	70
478	80	240	3	(40, 80, 120)			1307.9	1299.0	1307.9		*TL*	1	300	50
479	80	240	4	(60, 60, 60, 60)			1326.7	1322.7	1326.7		*TL*	1	300	10
480	80	240	4	(20, 30, 60, 130)			1277.2	1277.2	1277.2		*TL*	1	0	0
481	90	120	2	(60, 60)		1531.3	1519.7	1498.3	1514.2	0.8	*TL*	15	300	320
482	90	120	2	(40, 80)		1531.2	1517.2	1496.2	1512.9	0.9	*TL*	12	300	320
483	90	120	3	(40, 40, 40)		1576.5	1547.6	1531.2	1545.8	1.8	*TL*	2	300	320
484	90	120	3	(20, 40, 60)			1533.4	1519.6	1533.4		*TL*	1	300	231
485	90	120	4	(30, 30, 30, 30)			1566.9	1553.3	1566.9		*TL*	1	300	247
486	90	120	4	(10, 15, 30, 65)			1556.5	1548.2	1556.5		*TL*	1	300	80
487	90	120	2	(60, 60)	*	1559.6	1559.6	1528.5	1557.2	0.0	2032.9	13	300	320
488	90	120	2	(40, 80)		1562.8	1559.8	1528.4	1556.6	0.2	*TL*	19	300	320
489	90	120	3	(40, 40, 40)			1591.9	1561.3	1591.9		*TL*	1	300	312
490	90	120	3	(20, 40, 60)			1566.3	1538.9	1566.3		*TL*	1	300	254
491	90	120	4	(30, 30, 30, 30)			1626.7	1604.0	1626.7		*TL*	1	300	233

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
492	90	120	4	(10, 15, 30, 65)			1610.0	1592.5	1610.0		*TL*	1	300	95
493	90	120	2	(60, 60)		1466.2	1435.4	1410.7	1433.2	2.1	*TL*	7	300	320
494	90	120	2	(40, 80)		1458.0	1434.6	1410.7	1432.0	1.6	*TL*	8	300	320
495	90	120	3	(40, 40, 40)			1444.1	1426.2	1444.1		*TL*	1	300	291
496	90	120	3	(20, 40, 60)			1429.3	1413.0	1429.3		*TL*	1	300	243
497	90	120	4	(30, 30, 30, 30)			1497.3	1486.9	1497.3		*TL*	1	300	220
498	90	120	4	(10, 15, 30, 65)			1478.1	1468.0	1478.1		*TL*	1	300	54
499	90	120	2	(60, 60)		1473.9	1473.0	1446.1	1469.5	0.1	*TL*	25	300	320
500	90	120	2	(40, 80)		1470.6	1467.5	1445.7	1465.4	0.2	*TL*	24	300	320
501	90	120	3	(40, 40, 40)		1485.9	1482.5	1461.4	1480.8	0.2	*TL*	5	300	320
502	90	120	3	(20, 40, 60)			1475.8	1451.7	1475.8		*TL*	1	300	296
503	90	120	4	(30, 30, 30, 30)		1542.8	1536.8	1515.0	1535.8	0.4	*TL*	2	300	308
504	90	120	4	(10, 15, 30, 65)			1493.2	1476.2	1493.2		*TL*	1	300	162
505	90	120	2	(60, 60)		1889.7	1881.5	1851.0	1878.4	0.4	*TL*	17	300	320
506	90	120	2	(40, 80)		1899.0	1873.6	1833.8	1871.1	1.3	*TL*	8	300	320
507	90	120	3	(40, 40, 40)			1927.6	1910.4	1927.6		*TL*	1	300	236
508	90	120	3	(20, 40, 60)			1886.4	1875.2	1886.4		*TL*	1	300	136
509	90	120	4	(30, 30, 30, 30)			1953.9	1945.9	1953.9		*TL*	1	300	104
510	90	120	4	(10, 15, 30, 65)			1925.6	1915.9	1925.6		*TL*	1	300	43
511	90	240	2	(120, 120)			1244.2	1225.4	1244.2		*TL*	1	300	320
512	90	240	2	(80, 160)			1240.6	1225.4	1240.6		*TL*	1	300	295
513	90	240	3	(80, 80, 80)			1232.4	1227.6	1232.4		*TL*	1	300	21
514	90	240	3	(40, 80, 120)			1230.7	1227.3	1230.7		*TL*	1	300	10
515	90	240	4	(60, 60, 60, 60)			939.5	939.5	939.5		*TL*	1	0	0
516	90	240	4	(20, 30, 60, 130)							*TL*	0	0	0
517	90	240	2	(120, 120)			1178.4	1150.8	1178.4		*TL*	1	300	320
518	90	240	2	(80, 160)			1178.2	1149.6	1178.2		*TL*	1	300	305
519	90	240	3	(80, 80, 80)			1163.9	1152.8	1163.9		*TL*	1	300	40
520	90	240	3	(40, 80, 120)			1161.8	1151.2	1161.8		*TL*	1	300	20
521	90	240	4	(60, 60, 60, 60)			1161.8	1161.8	1161.8		*TL*	1	0	0
522	90	240	4	(20, 30, 60, 130)							*TL*	0	0	0
523	90	240	2	(120, 120)		1222.5	1188.9	1164.7	1187.9	2.8	*TL*	4	300	320
524	90	240	2	(80, 160)		1228.7	1188.6	1164.7	1187.8	3.3	*TL*	3	300	320
525	90	240	3	(80, 80, 80)			1185.8	1167.0	1185.8		*TL*	1	300	74
526	90	240	3	(40, 80, 120)			1181.9	1166.8	1181.9		*TL*	1	300	41
527	90	240	4	(60, 60, 60, 60)			1187.3	1184.3	1187.3		*TL*	1	300	0
528	90	240	4	(20, 30, 60, 130)							*TL*	0	0	0
529	90	240	2	(120, 120)			1017.2	1001.0	1017.2		*TL*	1	300	269
530	90	240	2	(80, 160)			1016.9	1000.5	1016.9		*TL*	1	300	224
531	90	240	3	(80, 80, 80)			1005.4	1001.0	1005.4		*TL*	1	300	20
532	90	240	3	(40, 80, 120)			1006.5	1001.0	1006.5		*TL*	1	300	10
533	90	240	4	(60, 60, 60, 60)							*TL*	0	0	0
534	90	240	4	(20, 30, 60, 130)							*TL*	0	0	0
535	90	240	2	(120, 120)			1193.3	1186.2	1193.3		*TL*	1	300	61
536	90	240	2	(80, 160)			1192.2	1186.3	1192.2		*TL*	1	300	70
537	90	240	3	(80, 80, 80)			1188.9	1188.9	1188.9		*TL*	1	0	0
538	90	240	3	(40, 80, 120)							*TL*	0	0	0
539	90	240	4	(60, 60, 60, 60)							*TL*	0	0	0
540	90	240	4	(20, 30, 60, 130)							*TL*	0	0	0
541	100	120	2	(60, 60)		1653.5	1649.6	1626.6	1646.4	0.2	*TL*	13	300	320
542	100	120	2	(40, 80)		1661.5	1646.7	1626.5	1644.0	0.9	*TL*	8	300	320
543	100	120	3	(40, 40, 40)			1690.8	1674.4	1690.8		*TL*	1	300	248
544	100	120	3	(20, 40, 60)			1655.0	1639.0	1655.0		*TL*	1	300	168
545	100	120	4	(30, 30, 30, 30)			1716.2	1698.4	1716.2		*TL*	1	300	164
546	100	120	4	(10, 15, 30, 65)			1716.4	1704.9	1716.4		*TL*	1	300	86
547	100	120	2	(60, 60)		2040.0	2001.2	1978.5	1999.6	1.9	*TL*	4	300	320
548	100	120	2	(40, 80)		2036.8	1997.5	1975.6	1996.9	2.0	*TL*	5	300	320
549	100	120	3	(40, 40, 40)			2011.3	1999.2	2011.3		*TL*	1	300	120
550	100	120	3	(20, 40, 60)			2000.2	1988.4	2000.2		*TL*	1	300	71
551	100	120	4	(30, 30, 30, 30)			2048.6	2043.4	2048.6		*TL*	1	300	42
552	100	120	4	(10, 15, 30, 65)			2019.1	2018.9	2019.1		*TL*	1	300	0
553	100	120	2	(60, 60)		1921.5	1899.8	1875.7	1899.5	1.1	*TL*	3	300	309
554	100	120	2	(40, 80)		1904.7	1899.0	1874.2	1896.3	0.3	*TL*	10	300	320
555	100	120	3	(40, 40, 40)			1970.6	1949.0	1970.6		*TL*	1	300	169
556	100	120	3	(20, 40, 60)			1917.3	1903.0	1917.3		*TL*	1	300	97
557	100	120	4	(30, 30, 30, 30)			1969.1	1958.2	1969.1		*TL*	1	300	73
558	100	120	4	(10, 15, 30, 65)			1961.3	1956.4	1961.3		*TL*	1	300	20
559	100	120	2	(60, 60)		1624.7	1609.5	1589.2	1608.0	0.9	*TL*	9	300	320
560	100	120	2	(40, 80)		1617.6	1597.8	1573.3	1595.3	1.2	*TL*	12	300	320
561	100	120	3	(40, 40, 40)			1636.8	1618.7	1636.8		*TL*	3	300	320
562	100	120	3	(20, 40, 60)			1632.0	1613.2	1632.0		*TL*	1	300	282
563	100	120	4	(30, 30, 30, 30)			1660.0	1647.2	1660.0		*TL*	1	300	177
564	100	120	4	(10, 15, 30, 65)			1651.4	1638.4	1651.4		*TL*	1	300	115
565	100	120	2	(60, 60)		1951.8	1927.0	1902.4	1921.3	1.3	*TL*	6	300	320
566	100	120	2	(40, 80)		1937.0	1924.0	1897.6	1917.9	0.7	*TL*	11	300	320

Continued on next page

No.	$\|V\|$	Q	m	$\mathbf{Q}$	opt	UB	LB_{tree}	LB_{LP}	LB_{cut}	gap	time	#BaB	#CC	#SRI
567	100	120	3	(40, 40, 40)	*	2025.4	2025.4	2005.1	2024.3	0.0	2206.2	3	300	294
568	100	120	3	(20, 40, 60)			1935.3	1918.8	1935.3		*TL*	1	300	261
569	100	120	4	(30, 30, 30, 30)		2133.6	2129.6	2113.1	2120.4	0.2	*TL*	6	300	320
570	100	120	4	(10, 15, 30, 65)			2119.5	2105.1	2119.5		*TL*	1	300	252
571	100	240	2	(120, 120)			1236.7	1217.7	1236.7		*TL*	1	300	151
572	100	240	2	(80, 160)			1236.0	1217.5	1236.0		*TL*	1	300	140
573	100	240	3	(80, 80, 80)			1218.5	1218.5	1218.5		*TL*	1	300	10
574	100	240	3	(40, 80, 120)			1219.3	1219.3	1219.3		*TL*	1	0	0
575	100	240	4	(60, 60, 60, 60)							*TL*	0	0	0
576	100	240	4	(20, 30, 60, 130)							*TL*	0	0	0
577	100	240	2	(120, 120)			1375.8	1361.0	1375.8		*TL*	1	300	102
578	100	240	2	(80, 160)			1376.4	1361.0	1376.4		*TL*	1	300	91
579	100	240	3	(80, 80, 80)			1362.2	1362.2	1362.2		*TL*	1	0	0
580	100	240	3	(40, 80, 120)			1366.8	1366.8	1366.8		*TL*	1	0	0
581	100	240	4	(60, 60, 60, 60)							*TL*	0	0	0
582	100	240	4	(20, 30, 60, 130)							*TL*	0	0	0
583	100	240	2	(120, 120)		1341.9	1291.6	1256.5	1291.6	3.8	*TL*	3	300	320
584	100	240	2	(80, 160)			1283.2	1256.5	1283.2		*TL*	1	300	316
585	100	240	3	(80, 80, 80)			1268.3	1256.9	1268.3		*TL*	1	300	51
586	100	240	3	(40, 80, 120)			1265.5	1256.9	1265.5		*TL*	1	300	30
587	100	240	4	(60, 60, 60, 60)			1278.5	1278.5	1278.5		*TL*	1	300	0
588	100	240	4	(20, 30, 60, 130)							*TL*	0	0	0
589	100	240	2	(120, 120)			1269.6	1253.6	1269.6		*TL*	1	300	90
590	100	240	2	(80, 160)			1268.6	1253.6	1268.6		*TL*	1	300	80
591	100	240	3	(80, 80, 80)			1257.9	1257.9	1257.9		*TL*	1	0	0
592	100	240	3	(40, 80, 120)							*TL*	0	0	0
593	100	240	4	(60, 60, 60, 60)							*TL*	0	0	0
594	100	240	4	(20, 30, 60, 130)							*TL*	0	0	0
595	100	240	2	(120, 120)			1379.7	1360.4	1379.7		*TL*	1	300	178
596	100	240	2	(80, 160)			1380.7	1360.4	1380.7		*TL*	1	300	186
597	100	240	3	(80, 80, 80)			1362.3	1362.3	1362.3		*TL*	1	300	0
598	100	240	3	(40, 80, 120)			1217.4	1217.4	1217.4		*TL*	1	0	0
599	100	240	4	(60, 60, 60, 60)			1373.9	1373.9	1373.9		*TL*	1	0	0
600	100	240	4	(20, 30, 60, 130)							*TL*	0	0	0