A New Exact Algorithm for Single-Commodity Vehicle Routing with Split Pickups and Deliveries

Submitted to *INFORMS Journal on Computing*
manuscript

A New Exact Algorithm for Single Commodity Vehicle Routing with Split Pickups and Deliveries

Jiliu Li
School of Management, Huazhong University of Science and Technology, China
lijiliu1992@outlook.com
Zhixing Luo
School of Management and Engineering, Nanjing University, China
luozx.hkphd@gmail.com
Roberto Baldacci
Engineering Management and Decision Sciences, College of Science and Engineering, Hamad Bin Khalifa University, Doha P.O. Box 34110, Qatar
rbaldacci@hbku.edu.qa
Hu Qin*
School of Management, Huazhong University of Science and Technology, China
tigerqin1980@qq.com
Zhou Xu
Faculty of Business, The Hong Kong Polytechnic University, Hong Kong
lgtzx@polyu.edu.hk

We present a new exact algorithm to solve a challenging vehicle routing problem with split pickups and deliveries, named as the Single Commodity Split Pickup and Split Delivery Vehicle Routing Problem (SPDVRP). In the SPDVRP, any amount of a product collected from a pickup customer can be supplied to any delivery customers, and the demand of each customer can be collected or delivered multiple times by the same or different vehicles. The vehicle fleet is homogeneous with limited capacity and maximum route duration. This problem arises regularly in inventory and routing rebalancing applications, such as in bike-sharing systems, where bikes must be rebalanced over time such that the appropriate number of bikes and open docks are available to users. The solution of the SPDVRP requires determining the number of visits to each customer, the relevant portions of the demands to be collected from or delivered to the customers, and the routing of the vehicles. These three decisions are intertwined, contributing to the hardness of the problem.

Our new exact algorithm for the SPDVRP is a branch-price-and-cut algorithm based on a pattern-based mathematical formulation. The algorithm relies on a novel label-setting algorithm used to solve the pricing problem associated with the pattern-based formulation, where the label components embed reduced cost functions, unlike those classical components that embed delivered or collected quantities, thus significantly reducing the dimension of the corresponding state space. Extensive computational results on different classes of benchmark instances illustrate that the newly proposed exact algorithm solves several open SPDVRP instances and significantly improves the running times of state-of-the-art algorithms.

Key words: vehicle routing, single commodity, split pickups, split deliveries, branch-price-and-cut, label-setting algorithm, exact algorithm.

1. Introduction

In this paper, we investigate a challenging vehicle routing problem with split pickups and deliveries, known as the Single Commodity Split Pickup and Split Delivery Vehicle Routing Problem (SPDVRP). Following the literature (Casazza et al. 2021), the problem can be defined as follows:

Consider a given digraph $G = (V, A)$, where $V = \{0, 1, \ldots, n, n+1\}$ is a vertex set and A is an arc set. Let $N = V \backslash \{0, n+1\}$ represent a set of n customers, and let vertex 0 represent a depot. For the sake of description convenience, we introduce a dummy depot $n' = n+1$ positioned at the same location as depot 0, and we regard depots 0 and n' as the start and end depots, respectively. The set of arcs A is defined as $A = \{(i,j) : i,j \in V,\ i \neq j,\ i \neq n+1,\ j \neq 0\} \setminus \{(0, n+1)\}$. Each vertex $i \in V$ has a demand q_i, which represents the amount of a product to be collected, if $q_i > 0$, or to be delivered, if $q_i < 0$. We assume $q_0 = q_{n'} = 0$ and that the problem is balanced, i.e., $\sum_{i \in N} q_i = 0$. As shown by Hernández-Pérez and Salazar-González (2004*a*), the *unbalanced* case, where $\sum_{i \in N} q_i \neq 0$, might be taken into account by adding a dummy customer positioned at the same location as depot 0 and having a demand equal to $-\sum_{i \in N} q_i$. Let $N^+ = \{i \ : \ i \in N, q_i > 0\}$ be the set of *pickup* customers and $N^- = \{i \ : \ i \in N, q_i < 0\}$ be the set of *delivery* customers. With each arc $(i,j) \in A$ is associated a routing or travel cost d_{ij} and a travel time t_{ij}, the latter including the service time at vertex i. We assume that matrices $[d_{ij}]$ and $[t_{ij}]$ satisfy the triangle inequality.

A fleet of identical vehicles having capacity Q stationed at depot 0 and represented by the index set $K = \{1, \ldots, |K|\}$ has to fulfill the customer demands. Each vehicle must start its route from vertex 0 and end at vertex n', and each route has a total duration not greater than a given limit T. We assume that each vehicle route departs from depot 0 and arrives at depot n' with an empty load, and we therefore also assume that the set of arcs A does not contain the arcs in $\{(0,i) : i \in N^-\} \cup \{(i,n') : i \in N^+\}$.

In the SPDVRP, any amount of product collected from a pickup customer can be supplied to any delivery customer, i.e., a *single commodity* is considered. The demand of each customer may be collected or delivered multiple times by the same or different vehicles, i.e., *split* deliveries or collections are allowed. In addition, it is assumed that temporary storage is not allowed (*non-preemptive* case) meaning that the demands cannot be temporarily dropped off at any customer.

The SPDVRP consists of designing at most $|K|$ routes of minimum total cost such that, subject to the vehicle capacity and duration constraints, the demand of all the customers is satisfied.

* Corresponding author

Based on the classification scheme proposed by Berbeglia et al. (2007), the SPDVRP is a *many-to-many*, *single-commodity*, *multi-vehicle* pickup and delivery problem `[M-M|P/D||`K`|]`. The SPDVRP is a generalization of the Split Delivery Vehicle Routing Problem (SDVRP) (Archetti and Speranza 2008, 2012, Irnich et al. 2014) in which the depot is the pickup location, the customers are delivery locations, and the demand of a customer is allowed to be served in more than one visit by different vehicles. The SPDVRP further extends the SDVRP by considering both pickup and delivery customers and maximum duration constraints in addition to vehicle capacity constraints. Because the SDVRP is $\mathcal{NP}$-hard, so is the SPDVRP.

The SPDVRP has practical applications in many routing problems especially in inventory repositioning, where the inventories of a product of a set of retailers must be rebalanced to fit the nature of the demand. Practical applications of the SPDVRP arise in the context of milk transportation, money transfer between branch offices of a bank (Hernández-Pérez and Salazar-González 2004*a*), self-service bike-sharing systems (Laporte et al. 2015, Espegren et al. 2016, Laporte et al. 2018), and transportation of crude oil (Izuno et al. 2012), to mention just a few. Important problems related to the SPDVRP are multi-commodity pickup and delivery problems, and comprehensive surveys of the different variants studied in the literature can be found in Berbeglia et al. (2007) and Parragh et al. (2008*a*,*b*). We also refer to Battarra et al. (2014) for a survey about pickup and delivery problems.

VRPs with split delivery are notoriously hard combinatorial optimization problems and the design of exact algorithms is not straightforward because the amount to be delivered at each customer visit is also a decision variable. State-of-the-art branch-price-and-cut (BPC) algorithms for classical non-split VRPs (e.g., see Costa et al. 2019) cannot be easily adapted to solve split delivery problems because handling the quantities to deliver either requires an exponential number of constraints in the master problem, or must be handled at the subproblem level, thus complicating the structure of the pricing algorithm.

In this paper, we present a new exact algorithm for the SPDVRP. The algorithm is developed from a pattern-based formulation of the problem where the LP-relaxation of the formulation is solved in a column generation fashion and embedded in an exact BPC solution approach. To solve the pricing problem associated with the pattern-based formulation, we develop a novel label-setting algorithm specifically designed for handling split deliveries. In this algorithm, label components embed reduced cost functions, unlike those classical components that embed delivered or collected quantities, thus significantly reducing the dimension of the corresponding state space. To further reduce the number of labels generated for speed up of the algorithm, we also derive and apply new dominance rules based on the reduced cost functions. We report extensive computational results

on different sets of benchmark instances, and we compare our results with the results of state-of-the-art algorithms for the SPDVRP. The results show that our new exact algorithm solves several open SPDVRP instances and significantly outperforms current state-of-the-art algorithms.

The remainder of this paper is organized as follows. The next section reviews problems related to the SPDVRP. §3 presents the pattern-based mathematical formulation of the SPDVRP. §4 describes the details of the procedure adopted to solve the LP-relaxation of the formulation, with emphasis on the definition of the reduced costs functions and the label-setting algorithm used to solve the pricing problem. Details of the exact algorithm are given in §5, followed by its computational evaluation in §6. Finally, we conclude the paper and indicate future research directions in §7.

2. Literature review

The SDVRP, a special case of the SPDVRP, has been addressed by many articles in the literature, most of them dealing with heuristic techniques, and surveys on the SDVRP and related problems can be found in Archetti and Speranza (2008, 2012) and in Irnich et al. (2014). The most recent exact algorithms for the SDVRP have been proposed by Archetti, Bianchessi and Speranza (2011) and Archetti et al. (2014). In Archetti, Bianchessi and Speranza (2011), a BPC algorithm is described for both the case where the fleet of vehicles is unlimited and the case where the fleet is limited to the minimum possible number of vehicles. Instances with up to 48 customers were consistently solved to optimality and an instance with 144 customers was also solved to optimality. Archetti et al. (2014) described two exact branch-and-cut (BC) algorithms based on two relaxed formulations and on procedures to obtain feasible solutions to the SDVRP from feasible solutions to the relaxed formulations. The results show that one of the two BC algorithms was capable solving most of the instances with up to 50 customers and two instances with 75 and 100 customers.

One of the most studied variant of the SDVRP is the SDVRP with Time Windows (SDVRPTW). Desaulniers (2010) described a BPC algorithm to solve the SDVRPTW. Archetti, Bouchard and Desaulniers (2011) enhanced the BPC algorithm of Desaulniers (2010) by introducing a tabu search algorithm to heuristically solve the pricing problem, several classes of valid inequalities and a new separation algorithm for the k-path inequalities. Instances with up to 100 customers were solved to optimality within a one-hour time limit. Recently, Luo et al. (2017) investigated a BPC algorithm for solving an extension of the SDVRPTW, called the SDVRPTW with linear weight-related cost (SDVRPTWL), in which the travel cost per unit distance is charged based on a linear function of the load weight. The algorithm was tested on both SDVRPTW and SDVRPTWL instances, and instances with up to 100 customers for both of the two variants were solved to optimality within a one-hour time limit. Bianchessi and Irnich (2019) presented a BC algorithm for the SDVRPTW. The algorithm is based on a relaxed compact model, in which some integer solutions are infeasible

for the SDVRPTW, and classes of valid inequalities are introduced to cut them. The computational experiments reported proved the optimality for several previously unsolved instances from the literature.

The SPDVRP is a generalization of the One-Commodity Pickup-and-Delivery Travelling Salesman Problem (1PDTSP) introduced in Hernández-Pérez and Salazar-González (2004*a*) and Hernández-Pérez and Salazar-González (2004*b*), where one-commodity and one capacitated vehicle is considered, and each customer must be visited exactly once. As for the SPDVRP, in the 1PDTSP any amount of product collected from a pickup customer can be supplied to any delivery customer. The SPDVRP is more challenging than the 1PDTSP since multiple vehicles and multiple visits to the customers are considered. Hernández-Pérez and Salazar-González (2004*a*) described a BC algorithm based on different valid inequalities which was further improved by Hernández-Pérez and Salazar-González (2007) with new valid inequalities, and which proved to be effective in solving to optimality instances with up to 100 customers. Salazar-González and Santos-Hernández (2015) further extended the 1PDTSP by introducing a more general problem, the Split-demand One-commodity Pickup-and-delivery Travelling Salesman Problem (1-SPDTSP). The 1-SPDTSP is a many-to-many, single-commodity, multi-vehicle problem, where the number of times that a vehicle visits a customer is limited and preemption is also allowed. The depot is considered as a standard customer having its own demand and capacity, and the vehicle is not forced to depart from or arrive at the depot empty. The problem has been modeled using a single-commodity flow formulation, and both a heuristic algorithm, based on the heuristic proposed by Hernández-Pérez and Salazar-González (2004*b*), and a BC approach, were described for its solution. The authors reported optimal solutions for 1-SPDTSP instances with up to 50 customers. Recently, Hernández-Pérez et al. (2018) have described a matheuristic algorithm that iteratively applies a constructive procedure and a refinement procedure with the aim of solving large-sized 1-SPDTSP instances with up to 500 customers.

Problems that are closely related to the SPDVRP and corresponding applications arise in the context of self-service bike-sharing systems. Due to the growth in popularity in recent years of self-service bike-sharing and scooter-sharing systems, there is a rapidly growing amount of literature on these problems. The survey paper of Laporte et al. (2015), and its updated version of Laporte et al. (2018), classify the relevant literature by considering station location, fleet dimensioning, station inventory, rebalancing incentives, and vehicle repositioning. Below, we briefly review the most recent exact algorithms in this area, and the reader is referred to the surveys of Laporte et al. (2015, 2018) for an in-depth analysis of the literature.

In the *static* variant of these problems it is assumed that bicycles are not moved by users and that capacitated vehicles must be used to reallocate the bicycles between the different stations

or locations. This corresponds to the practical scenario where every night the bicycle inventories at the locations must be restored to fit the demand at minimum travel cost. The problems are then classified according to (i) the number of maximum visits allowed to the stations (i.e., *single* or *multiple visits*), (ii) the *preemptive* or *non-preemptive* cases, and (iii) the number of vehicles available (*single* or *multiple vehicles*). In the *multiple visits* case, a station can be visited more than once by different vehicles, and a station is also allowed to be visited multiple times by the same vehicle. Two special cases of the multiple visits case arise: (i) a vehicle is allowed to visit a customer at most once, hereafter referred as a *multiple-visit, different-vehicle* case, and (ii) a customer can be visited (even more than once) by only one vehicle, hereafter referred as a *multiple visits, same vehicle* case.

Static, single-visit, non-preemptive problems were investigated by Erdoğan et al. (2014) and Dell'Amico et al. (2014). Erdoğan et al. (2014) addressed a single vehicle variant where stations might be visited at most once and the resulting numbers of bikes at stations after the rebalancing should lie within a given interval instead of exactly corresponding to a fixed target value. The objective function includes a commodity handling cost. The authors developed an integer programming formulation and described valid inequalities that have been used to develop a BC algorithm as well as a Benders decomposition scheme. The BC algorithm was capable of solving instances with up to 50 locations. Dell'Amico et al. (2014) considered a multiple vehicle problem, called the Bike Rebalancing Problem (BRP) where initially balanced stations are to be visited as well, to ensure a daily inspection. The authors described four mathematical formulations for the BRP and proposed a BC algorithm for its solution capable of solving to optimality BRP instances involving up to 50 locations or stations. Dell'Amico et al. (2016) solved the BRP by developing a destroy and repair heuristic, improving the best-known solutions for several instances from the literature.

Chemla et al. (2013) , Erdoğan et al. (2015), Casazza et al. (2019), Bulhões et al. (2018), Bruck et al. (2019) investigated static and multiple-visit variants. Chemla et al. (2013) studied a single-vehicle, preemptive problem. The authors proposed a mathematical formulation from which two relaxations are derived. A tabu search algorithm is also described to solve instances involving up to 100 stations. The same problem was also considered by Cruz et al. (2017), who obtained improved results on the existing benchmark instances by means of an iterated local search algorithm. Erdoğan et al. (2015) presented a BC algorithm for the preemptive single-vehicle variant, and reported results of computational tests on benchmark instances from the literature showing that instances with up to 60 stations can be solved to optimality in less than 2 hours of computing time. Casazza et al. (2019) considered a multiple-vehicle non-preemptive problem, called the Multiple Vehicle Balancing Problem (MVBP), with an additional constraint imposing a maximum number of visits per station. Based on theoretical properties of the problem, the authors proposed an integer linear

Table 1 Features of the main works related to the SPDVRP

Work	Visits	Vehicles	Type of visits	Preemptive	Constraints
	(s)ingle (m)ultiple (o)ptional	(s)ingle (m)ultiple	(sv) same veh. (df) diff. veh. (sv,dv)	(y)es (n)o	(c)apacity (m)ax. num. of visits (d)duration
SPDVRP	m	m	sv, dv	n	c, d
Hernández-Pérez and Salazar-González (2004*a*)	s	s	-	n	c
Chemla et al. (2013)	m	s	sv	y	c
Erdoğan et al. (2014)	o	s	sv	n	c
Dell'Amico et al. (2014)	s	m	-	n	c
Erdoğan et al. (2015)	m	s	sv	y	c
Salazar-González and Santos-Hernández (2015)	m	s	sv	y	c, m
Bulhões et al. (2018)	m	m	sv	n	c, m, d
Casazza et al. (2019)	m	m	sv, dv	n	c, m
Bruck et al. (2019)	m	s	sv	n	c
Casazza et al. (2021)	m	m	sv, dv	n	c, d

programming formulation and introduced some valid inequalities that were used to compute valid lower bounds in a column generation fashion. Upper bounds were also computed by means of a rounding heuristic and a Memetic algorithm and the lower and upper bounds computed were used to solve to proven optimality instances with up to 20 stations. Any instance of the MVBP can be transformed into an equivalent SPDVRP instance simply by modeling the additional constraint on the maximum number of visits (say α), by setting the travel time $t_{ij} = 1$, $\forall (i,j) \in A$, and $T = \alpha$. Bulhões et al. (2018) investigated a multiple-visit, multiple-vehicle, non-preemptive problem with an upper limit on the maximum number of visits to a station. In addition, a station could only be served (even multiple times) by one of the fleet vehicles, and the vehicle workload constraint was also considered. The authors designed a BC algorithm based on an extended network-based mathematical formulation and an iterated local search-based heuristic. The BC was capable of solving to optimality several instances with 20 stations and 1 instance with 30 stations and 3 vehicles within the time limit of one hour. Bruck et al. (2019) also investigated a single-vehicle, non-preemptive problem. The authors investigated theoretical results concerning problem complexity and worst-case analysis, and then proposed three exact algorithms based on different mathematical formulations. Computational experiments were reported for instances involving up to 80 stations.

To the best of our knowledge, the only work considering the SPDVRP can be found in Casazza et al. (2021). The authors proposed a formulation where routes are decomposed into sequences of simpler substructures called *clusters*. Valid inequalities, a rounding heuristic, and a branch-and-price (BP) algorithm are also described. The proposed algorithm is competitive with the algorithm proposed in Casazza et al. (2019) for the MVBP, and instances with up to 20 stations were solved to optimality. In the computational results in Section 6, we compare our results with the results reported in Casazza et al. (2019) and Casazza et al. (2021).

Table 1 provides a summary of the main problem features of works closely related to the SPDVRP. From the table, it is seen that the SPDVRP addresses the most general case in terms of

types of visit to the customers, which is regarded as the main factor that influences the complexity of split delivery problems.

3. A pattern-based formulation for the SPDVRP

The pattern-based (PB) formulation is based on the set partitioning formulation originally proposed by Balinski and Quandt (1964) for the Capacitated VRP (CVRP), and it associates a decision variable with each feasible route of the SPDVRP.

A *route* in graph G is defined as a (not necessarily elementary) path $R=(0,i_1,\ldots,i_b,n')$ starting at depot 0, ending at depot n', visiting a set of vertices $\{i_1,\ldots,i_b\}\subseteq N$ (maybe more than once), and being such that the total travel time of the route is less than or equal to T. Associated with a route R are (i) the set of *quantities* $\{l_{i_1},\ldots,l_{i_b}\}$ delivered (negative values) and collected (positive values) to the visited vertices, and (ii) a *demand pattern*, or simply *pattern* $\alpha\in\mathbb{R}^{|N|}$, where α_i for $i\in N^+$ with $0\leq\alpha_i\leq q_i$ represents the total amount of the product picked up from customer i, and α_i for $i\in N^-$ with $q_i\leq\alpha_i\leq 0$ represents the total quantity delivered to customer i. Note that if the route is elementary, for a customer $i\in N$ we have $\alpha_i=l_i$ if $i\in\{i_1,\ldots,i_b\}$, and $\alpha_i=0$ otherwise.

A set of quantities and a demand pattern are feasible for route $R=(0,i_1,\ldots,i_b,n')$, and we simply refer to it as a *feasible pattern*, if (i) the vehicle starts and ends the route R with an empty load, (ii) along the route R the total demand collected from the pickup customers is delivered to the delivery customers, i.e., $\sum_{i\in N}\alpha_i=0$, and (iii) the load of the vehicle after it visits the h-th vertex of the route R is nonnegative and does not exceed the vehicle capacity Q, i.e., $0\leq\sum_{s=1}^{h}l_{i_s}\leq Q$, $h=1,\ldots,b-1$.

Let $\mathcal{R}$ be the index set of all routes where the route of index $r\in R$ is denoted by R_r. For each $r\in R$, let $\mathcal{P}_r$ be the index set of all feasible patterns associated with route R_r. Given a route $r\in\mathcal{R}$ and a pattern index $p\in\mathcal{P}_r$, let α_{ipr} be a continuous coefficient equal to the total demand collected from or delivered to customer i by route R_r according to the pattern with index p. In addition, we denote by c_r the routing cost associated with route R_r, computed as $c_r=\sum_{(i,j)\in A}\gamma_{ijr}d_{ij}$ where γ_{ijr} is the number of times that arc (i,j) is traversed by route R_r. Due to the constraint on the maximum route duration, each route R_r satisfies that $\sum_{(i,j)\in A}\gamma_{ijr}t_{ij}\leq T$.

Let θ_{pr} be a nonnegative integer variable representing the number of vehicles assigned to pattern $p\in\mathcal{P}_r$ of route R_r for $r\in\mathcal{R}$. Formulation PB is as follows:

$$(PB)\quad \min \sum_{r\in\mathcal{R}}\sum_{p\in\mathcal{P}_r} c_r\theta_{pr} \tag{1a}$$

$$\text{s.t.} \sum_{r\in\mathcal{R}}\sum_{p\in\mathcal{P}_r} \alpha_{ipr}\theta_{pr}\leq q_i,\quad \forall\, i\in N, \tag{1b}$$

$$\sum_{r\in\mathcal{R}}\sum_{p\in\mathcal{P}_r} \theta_{pr}\leq |K|, \tag{1c}$$

$$\theta_{pr}\geq 0 \text{ and integer},\quad \forall\, p\in\mathcal{P}_r, r\in\mathcal{R}. \tag{1d}$$

The objective function (1a) states to minimize the total routing cost. Constraints (1b) impose that the demand of each customer is satisfied by the demand patterns associated with the routes selected in the solution, and their correctness arise from the fact that $\sum_{i \in N} q_i = 0$. Constraints (1c) limits the number of vehicles used to the number of vehicles available.

Casazza et al. (2019) also described a pattern-based formulation to model the MVBP. However, the authors assume that $|q_i| \leq Q$, $\forall i \in N$, and showed that under this assumption variables θ_{pr} can be stated as binary variables. To remove the limit $|q_i| \leq Q$, $i \in N$, their algorithm requires to introduce $k = \lceil |q_i|/Q \rceil$ vertices for vertex i, say $\{i_1, \ldots, i_k\}$, such that $|q_{i_h}| = Q$, $h = 1, \ldots, k-1$, and $|q_{i_k}| = |q_i| - Q(K-1)$.

The LP-relaxation of formulation PB can be solved in a column generation fashion by iteratively solving a *restricted master problem* (RMP) and the pricing problem, which determines whether there exists a variable θ_{pr} to be added to the RMP to improve its current solution. For a thorough description of the column generation technique and corresponding solution approaches, the reader is referred to the book of Desaulniers et al. (2005) and the reviews of Barnhart et al. (1998) and Lübbecke and Desrosiers (2005).

3.1. Pricing problem

Let $\mu = (\mu_0, \mu_1, \mu_2, \ldots, \mu_n)$, where $\mu_1, \mu_2, \ldots, \mu_n \leq 0$, and $\mu_0 \leq 0$ are the dual variables associated with constraints (1b) and (1c), respectively. Given a route $r \in \mathcal{R}$ represented by path $R_r = (0, i_1, \ldots, i_b, n')$, the most negative reduced cost pattern α associated with the route can be computed by solving the following LP problem:

$$c_r - \mu_0 + min\left(-\sum_{i \in V(R_r)} \alpha_i \mu_i\right)$$

$$\text{s.t.} \quad 0 \leq \sum_{s=1}^{h} l_{i_s} \leq Q, \quad h = 1, \ldots, b-1, \tag{2a}$$

$$\sum_{i \in V(R_r)} \alpha_i = 0, \tag{2b}$$

$$0 \leq \alpha_i = \sum_{s \in R_r(i)} l_{i_s} \leq q_i, \quad \forall\ i \in V(R_r) \cap N^+, \tag{2c}$$

$$q_i \leq \alpha_i = \sum_{s \in R_r(i)} l_{i_s} \leq 0, \quad \forall\ i \in V(R_r) \cap N^-, \tag{2d}$$

$$0 \leq l_{i_s} \leq q_i, \quad \forall s \in R_r(i), i \in N^+, \tag{2e}$$

$$q_i \leq l_{i_s} \leq 0, \quad \forall s \in R_r(i), i \in N^-, \tag{2f}$$

$$\alpha_i = 0, \quad \forall i \in N \setminus V(R_r), \tag{2g}$$

where $V(R_r)$ is the set of customers visited by route R_r and $R_r(i) \subset \{1, 2, \ldots, b\}$ for $i \in V(R_r)$, is the set of vertex indices of route R_r corresponding to customer $i \in N$, i.e., $i_h = i_k$, $\forall i, k \in R_r(i)$,

$h \neq k$. In the formulation, constraints (2a) impose the vehicle capacity constraints, and constraints (2e) and (2f) are redundant due to the definition of variables α_i given by constraints (2c) and (2d). It can be seen that the formulation admits a feasible solution where each customer $i \in V(R_r)$ can be associated with $\alpha_i = 0$.

Based on the above formulation, the pricing problem therefore calls for the joint definition of route R and the associated pattern α, a variant of the shortest path problem with resource constraints (SPPRC) (Irnich and Desaulniers 2005, Irnich and Villeneuve 2006) that allows for the generation of paths with cycles (i.e., routes with multiple visits to the same customer). The SPPRC is a relaxation of the elementary SPPRC (ESPPRC), a strongly $\mathcal{NP}$-hard problem, where elementarity requirements are also considered. The ESPPRC was studied in the context of vehicle routing problems by several researchers (see Feillet et al. 2004, Chabrier 2006, Righini and Salani 2006, 2008), who developed effective dynamic programming algorithms for solving it. The SPPRC is easier to solve than the ESPPRC, as it can be solved by a pseudo-polynomial algorithm, and a number of SPPRC relaxations have been proposed in the VRP literature (see Costa et al. 2019).

As shown by the literature, the presence of the split deliveries and collections further complicates the ESPPRC (Desaulniers 2010, Archetti, Bouchard and Desaulniers 2011, Casazza et al. 2021) and the SPPRC (Casazza et al. 2019). Indeed, in these variants the quantity delivered or collected at each visited customer is a decision variable.

The algorithm of Casazza et al. (2019) requires a state variable for each customer that accounts for the total number of units picked up or delivered at the customer. The algorithm follows the procedure proposed by Baldacci et al. (2008) for the Pickup and Delivery Problem with Time Windows where forward paths are first generated and combined to form complete (not necessarily elementary) routes.

The pricing algorithm of Desaulniers (2010) solves an ESPPRC and exploits certain properties of the pricing problem related to the Knapsack Problem. The algorithm requires state variables indicating (i) the total quantity delivered in the full deliveries, (ii) whether or not a split delivery occurred, (iii) the maximum quantity that can be delivered in the split delivery and (iv) the unit dual price for the split delivery. In this way, when extending a label from a vertex i to a vertex j, their pricing algorithm creates up to three labels associated with a zero delivery, a split delivery and a full delivery to j, respectively. The pricing algorithm relies on the use of reduced (linear) cost functions. More precisely, a state variable is used to compute the maximal reduced cost that is obtained by replacing the split delivery, if any, by a zero delivery. The reduced cost function of a label (and associated path) is then defined as $c - \pi(x - L)$, where: (i) c is the maximal reduced cost, (ii) π is the unit dual price of the split delivery, (iii) L is the total quantity delivered in the full deliveries, and (iv) $x \in [L, L + l]$ is the total quantity delivered, including the split delivery if

any, where l is the maximum quantity that can be delivered in the split delivery. Given the reduced cost function of a label, dominance rule designed for classical ESPPRC (Irnich and Desaulniers 2005) cannot be used directly. Desaulniers (2010) therefore described new dominance rules based on the comparison of the reduced cost functions associated with two labels.

The algorithm of Casazza et al. (2021) is inspired to the one proposed by Desaulniers (2010). Casazza et al. (2021) solves the pricing problem of an alternative SPDVRP formulation where routes are decomposed into sequences of simpler substructures called *clusters*. Their pricing algorithm exploits properties of the pricing problem related to the Fractional Knapsack Problem with Penalties and, similarly to Desaulniers (2010), requires state variables representing the residual capacity of a partial path and the existence (in the partial path) of a fractional vertex, that is, a vertex having its demands fractionally serviced.

The label-setting algorithm we propose also relies on reduced cost functions, but they are expressed as general piecewise-linear convex functions and embedded as components for the labels. The main advantage is that a label can be defined without the use of components related to the quantities delivered or collected, thus reducing the dimension of the state space. Furthermore, we propose some set-dominance rules used to reduce the number of labels, and therefore increase the performance of the label-setting algorithm.

The use of general functions as label attributes has also been adopted by Liberatore et al. (2011) for the VRP with soft time windows, by Dabia et al. (2013) for solving the time-dependent VRPTW, and by Luo et al. (2017) for the SDVRPTWL. The label-setting algorithms of Liberatore et al. (2011) and Luo et al. (2017) use piecewise linear reduced cost functions of the start of service time and of the quantity delivered, respectively, whereas in Dabia et al. (2013) the function represents the ready time at the last node visited by the label as a function of the departure time at the depot.

3.2. Relaxation $\overline{PB}$

To overcome these drawbacks and in order to reduce the complexity of the pricing problem, we consider a relaxation of formulation PB where the index set of patterns $\mathcal{P}_r$ of route R_r for $r \in \mathcal{R}$ is enlarged to set $\overline{\mathcal{P}}_r$ containing also indices of patterns where the total demand collected from or delivered to a customer can exceed the demand of the customer, i.e., coefficients α_{ipr} can exceed the demand associated with customer i (either positively or negatively). More specifically, we relax constraints (2c) and (2d) of the above formulation defining a feasible pattern. In the new pattern definition, the amount of the product collected from or delivered to a customer each time the customer is visited cannot exceed the customer demand, as imposed by constraints (2e) and (2f). We denote by $\overline{PB}$ the formulation obtained by substituting the index set of patterns $\mathcal{P}_r$, $\forall r \in \mathcal{R}$,

with $\overline{\mathcal{P}}_r$. It is worth noting that formulation $\overline{PB}$ under the integrality requirements (1d) provides a correct SPDVRP formulation. In the following, we denote by $\overline{LPB}$ the LP-relaxation of formulation $\overline{PB}$.

4. Solving the pricing problem of formulation $\overline{LPB}$

This section describes the algorithm adopted to solve the pricing problem associated with formulation $\overline{LPB}$, which is a key component of our exact algorithm.

Below we give the details of the reduced cost functions used together with their properties, followed by a description of the label-setting algorithm.

4.1. Forward, backward paths and reduced cost functions

We define a *forward path* $P = (i_0 = 0, i_1, \ldots, i_{s-1}, i_s)$ as a (not necessarily elementary) path starting at depot 0, visiting vertices $V(P) = \{0, i_1, \ldots, i_{s-1}, i_s\}$ and ending at customer i_s with a total duration $t(P)$ less than or equal to T. Similarly, a *backward path* $\overline{P} = (i_s, i_{s-1}, \ldots, i_1, i_0 = n')$ is a (not necessarily elementary) path starting at vertex i_s, visiting vertices $V(\overline{P}) = \{i_s, i_{s-1}, \ldots, i_1, 0\}$ and ending at the depot n' with a total duration $t(\overline{P})$ less than or equal to T.

The algorithm is based on the following observations:

(i) Any route R represented by path $R = (0, i_1, \ldots, i_b, n')$ can be decomposed, for every $i = i_s$, $s \in \{1, \ldots, b-1\}$, into a forward path P ending at i_s and a backward path $\overline{P}$ starting at $j = i_{s+1}$ such that $t(P) + t_{ij} + t(\overline{P}) \leq T$.

(ii) The set of patterns α associated with route R satisfying constraints (2a), (2b), (2e), (2f) and (2g), can be partitioned by the amount w of demand picked up from the pickup customers of path P and left for the delivery customers of path $\overline{P}$, that is to say that the vehicle load along arc (i, j) joining paths P and $\overline{P}$ is equal to w.

(iii) Let $\overline{c}^f_P : \mathbb{R}_+ \to \mathbb{R}$ be a *forward reduced cost function*, where $\overline{c}^f_P(w)$ is equal to the cost of the minimum reduced cost path P and associated demand pattern $\alpha(P)$ if a quantity w is delivered to the customers of path $\overline{P}$. In addition, let $\overline{c}^b_{\overline{P}} : \mathbb{R}_+ \to \mathbb{R}$ be a *backward reduced cost function*, where $\overline{c}^b_{\overline{P}}(w)$ is equal to the cost of the minimum reduced cost path $\overline{P}$ and associated demand pattern $\alpha(\overline{P})$ if a quantity w is collected from the customers of path P. The reduced cost $\overline{c}(R)$ of the route and corresponding demand pattern having minimum reduced cost can be computed as:

$$\overline{c}(R) = \min_{0 \leq w \leq \min\{W(P), W(\overline{P})\}} \{\overline{c}^f_P(w) + d_{ij} + \overline{c}^b_{\overline{P}}(w)\},$$

where $W(P) = \min\{Q, \sum_{i \in V(P): i \in N^+} q_i\}$ is the maximum amount of product collected from the pickup customers along path P and $W(\overline{P}) = \min\{Q, \sum_{i \in V(\overline{P}): i \in N^-} -q_i\}$ is the maximum amount of the product required by the delivery customers along path $\overline{P}$.

The following theorems state that functions $\overline{c}_P^f$ and $\overline{c}_P^b$ admit piecewise-linear convex representations. For the sake of the exposition, we consider a generic piecewise-linear convex function $f : \mathbb{R}_+ \to \mathbb{R}$ expressed as $f(x) = \max_{h \in H}\{a_h x + b_h\}$, where $H = \{1, 2, \ldots, m\}$ is the index set of its *breakpoints*.

THEOREM 1. *Function $\overline{c}_P^f$ is a piecewise-linear convex nondecreasing function.*

Proof. The proof is provided in the e-companion to this paper (see EC.1.1). □

THEOREM 2. *Function $\overline{c}_P^b$ is a piecewise-linear convex nonincreasing function.*

Proof. The proof is provided in the e-companion to this paper (see EC.1.2). □

Based on the above observations, our pricing algorithm considers in each label a piecewise linear reduced cost function of the quantity w. The functions are properly initialized and updated with each extension. When performing an extension along an arc (i, j), the reduced cost function of the new label is obtained by summing up the function from the previous label and the function associated with the delivery or pickup service at j. The various components of the algorithm are described in detail below.

4.2. A label-setting algorithm for the pricing problem

The algorithm is based on the *bidirectional search* algorithm proposed by Righini and Salani (2006), where *forward* and *backward* paths are first generated and then combined to form complete routes by a *joining procedure*.

Forward extension A forward path $P = (i_0 = 0, i_1, \ldots, i_{s-1}, i_s)$ is represented by a label L^f associated with the last vertex i_s where $L^f = (v^f, s^f, V^f, W^f, \{(a_h^f, b_h^f)\}_{h \in H^f})$ is composed of the following information:

- $v^f \in V$: the last vertex visited.
- s^f: the total duration.
- $V^f \subseteq V$: the set of vertices that could be reached from the last vertex v^f.
- W^f: the maximum amount of product that can be delivered to the delivery customers to be appended to the path.
- $\{(a_h^f, b_h^f)\}_{h \in H^f}$: the representation of the piecewise-linear convex nondecreasing function.

In the label L^f, for a given $w \in [0, W^f]$ representing the demand picked up by path P and left for the vertices to be appended to vertex v^f, the reduced cost associated with the path can be computed as $\max_{h \in H^f}\{a_h^f w + b_h^f\}$.

A forward label L_1^f with $i = v_1^f$ can be extended to a vertex $j \in V_1^f$ yielding a new label L_2^f based on the following extension functions. We have two cases:

(i) $j \in N^+$ (pickup customer). The components of label L_2^f are computed as follows:

- $v_2^f = j$.
- $s_2^f = s_1^f + t_{ij}$.
- $V_2^f = V_1^f \setminus \{l : (j,l) \in A, s_2^f + t_{jl} + t_{ln'} > T\} \setminus \{j\}$.
- $W_2^f = \min\{Q, W_1^f + q_j\}$.
- $$\{(a_h^f, b_h^f)\}_{h \in H_2^f} = \left\{(a_h^f, \ b_h^f + d_{ij})\right\}_{h \in H_1^{f-}} \bigcup \left\{(a_h^f, b_h^f - (a_h^f + \mu_j)q_j + d_{ij})\right\}_{h \in H_1^{f+}} \bigcup \left\{(-\mu_j, \max\left\{z_1^*, \max_{h \in H_1^{f-}}\{(a_h^f + \mu_j)W_1^f + b_h^f\}, \max_{h \in H_1^{f+}}\{b_h^f\}\right\} + d_{ij})\right\},$$

where $H_1^{f-} = \{h \in H_1^f : a_h^f + \mu_j < 0\}$, $H_1^{f+} = \{h \in H_1^f : a_h^f + \mu_j \geq 0\}$ and $z_1^* = \min\{\max_{h \in H_1^f}\{a_h^f x + b_h^f\}\}$.

(ii) $j \in N^-$ (delivery customer). Components v_2^f, s_2^f and V_2^f are computed as in the previous case, whereas the remaining components are updated as follows:

- $W_2^f = W_1^f$.
- $$\{(a_h^f, b_h^f)\}_{h \in H_2^f} = \left\{(a_h^f, b_h^f - (a_h^f + \mu_j)q_j + d_{ij})\right\}_{h \in H_1^{f-}} \bigcup \left\{(a_h^f, b_h^f + d_{ij})\right\}_{h \in H_1^{f+}} \bigcup \left\{(-\mu_j, \max\left\{z_1^*, \max_{h \in H_1^{f-}}\{(a_h^f + \mu_j)W_1^f + b_h^f\}\right\} + d_{ij})\right\},$$

where H_1^{f-}, H_1^{f+} and z_1^* are defined as in the previous case.

In the expansion, label W^f is updated only for vertices in N^+, and its upper limit Q represents the maximum quantity that can be picked up and left for the vertices to be appended to the path. Note that the update of the reduced cost functions follows the scheme of the proof of Theorem 1.

The effectiveness of a label-setting algorithm strongly relies on the use of dominance rules aimed at removing dominated paths that cannot be part of any optimal solution. In the above definition, with the label L^f, each path P is associated with a set of infinite loading patterns defined by values $w \in [0, W^f]$. As illustrated by Irnich and Desaulniers (2005), given two labels L_1^f and L_2^f, respectively, label L_1^f is dominated by label L_2^f if:

(i) $v_1^f = v_2^f$.

(ii) Any feasible extension of label L_1^f is also feasible for L_2^f.

(iii) Extending label L_1^f always results in a route having a reduced cost greater than or equal to the reduced cost of the route obtained from label L_2^f and the extension of L_1^f.

However, verifying the above conditions is not straightforward. Given values $w_1 \in [0, W_1^f]$ and $w_2 \in [0, W_2^f]$, a sufficient condition for L_2^f to dominate L_1^f is

(i) $v_1^f = v_2^f$.

(ii) $s_2^f \leq s_1^f$.

(iii) $V_2^f \subseteq V_1^f$.

(iv) $w_2 = w_1$.

(v) $\max_{h\in H_2^f}\{a_h^f w_2 + b_h^f\} \le \max_{h\in H_1^f}\{a_h^f w_1 + b_h^f\}$.

For our algorithm, such dominance rule is not applicable, because the quantities w_1 and w_2 are not components of the labels. Nevertheless, given two labels L_1^f and L_2^f such that conditions (i)-(iii) hold, L_1^f is dominated by label L_2^f if $W_2^f \ge W_1^f$ and $\max_{h\in H_2^f}\{a_h^f w + b_h^f\} \le \max_{h\in H_1^f}\{a_h^f w + b_h^f\}$, for all $w \in [0, W_1^f]$. In practice, the latter dominance rule can be quite weak since it requires that the reduced cost function of L_2^f lies below the one of L_1^f on the interval $[0, W_1^f]$. We therefore extend the dominance rule to compare L_1^f with a set of labels for which we compute the lower envelope of the set of piecewise-linear convex functions associated with the labels. The dominance rule, called a *set-dominance* rule, is defined as follows.

Let $L_1^f = (v_1^f, s_1^f, V_1^f, W_1^f, \{(a_h^f, b_h^f)\}_{h\in H_1^f})$, and define $\mathscr{L}_1^f$ to be the set of labels $L^f = (v^f, s^f, V^f, W^f, \{(a_h^f, b_h^f)\}_{h\in H^f})$ such that (i) $v^f = v_1^f$, (ii) $s^f \le s_1^f$, and (iii) $V_1^f \subseteq V^f$.

Given a value w with $0 \le w \le Q$, for a label $L^f \in \mathscr{L}_1^f$ define

$$g_{L^f}(w) = \begin{cases} \max_{h\in H^f}\{a_h^f w + b_h^f\}, & w \le W^f, \\ +\infty, & W^f < w \le Q, \end{cases} \tag{3}$$

and for the set of labels $\mathscr{L}_1^f$ define

$$\overline{g}_{\mathscr{L}_1^f}(w) = \min_{L^f\in\mathscr{L}_1^f}\{g_{L^f}(w)\}.$$

Function $\overline{g}_{\mathscr{L}_1^f}(.)$ computes the lower envelope of the set of reduced cost functions associated with the label set $\mathscr{L}_1^f$. The following dominance can be used to reduce the number of labels.

Dominance 1 (Forward set-dominance) *Label L_1^f is dominated by the label set $\mathscr{L}_1^f$ if $g_{L_1^f}(w) \ge \overline{g}_{\mathscr{L}_1^f}(w)$, $\forall w$, $0 \le w \le Q$.*

Proof. The proof is provided in the e-companion to this paper (see §EC.1.3). □

Figure 1 illustrates the use of Dominance 1. Figure (a) depicts functions $g_{L_x^f}(.)$, $x = 1, 2, 3$, associated with three labels L_1^f, L_2^f, and L_3^f, respectively, with $\mathscr{L}_1^f = \{L_2^f, L_3^f\}$. The figure shows that label L_1^f is dominated by neither label L_2^f nor by label L_3^f in the interval $[0, W_1^f]$. The plot of function $\overline{g}_{\mathscr{L}_1^f}(w)$ given by figure (b) clearly shows that the function is always inferior to function $g_{L_1^f}(.)$ in the interval $[0, W_1^f]$, thus label L_1^f is dominated by the combination of labels L_2^f and L_3^f, and can be safely discarded.

Forward labels are generated by starting from the initial label $L^f = (0, \{0\}, 0, N, 0, \{(0, -\nu_0)\})$. In addition, as described for bidirectional labeling by Righini and Salani (2006), time is considered to be the critical resource, and labels with a capacity consumption larger than or equal to $\lceil T/2 \rceil$ are not generated.

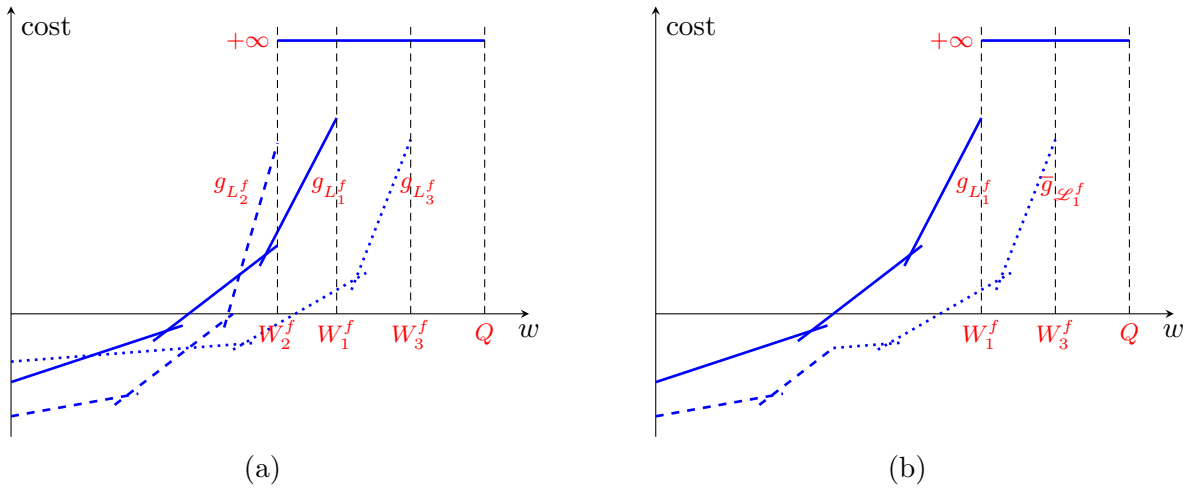

Figure 1 **Illustration of the forward set-dominance defined in Dominance 1**

To check the dominance rule, we associate with a pair (i,s), $i \in N$, $1 \le s \le T$, a *bucket* $\mathscr{F}_i(s)$ containing all non-dominated labels $L^f = (v^f, s^f, \{(a_h^f, b_h^f)\}_{h \in H^f})$, with $v^f = i$ and $s^f = s$. Each bucket $\mathscr{F}_i(s)$ has associated a function $\overline{g}_{\mathscr{F}_i(s)}(w) = \min_{L^f \in \mathscr{L}(s)}\{g_{L^f}(w)\}$ (see also (3)), where $\mathscr{L}(s) = \{L^f : L^f \in \mathscr{F}_i(s'), s' \le s\}$. Whenever a new label L^f is generated, the label is checked for Dominance 1 on bucket $\mathscr{F}_{v^f}(s^f)$, that is, if $g_{L^f}(w) \ge \overline{g}_{\mathscr{F}_{v^f}(s^f)}$, $\forall w$, $0 \le w \le Q$, label L^f is dominated. If the label is non-dominated, it is inserted in bucket $\mathscr{F}_{v^f}(s^f)$ and the functions of bucket $\mathscr{F}_{v^f}(s^f)$ and of all buckets $\mathscr{F}_{v^f}(s)$, $s > s^f$, are updated as follows:

$$\overline{g}_{\mathscr{F}_{v^f}(s)}(w) = \min\left\{\overline{g}_{\mathscr{F}_{v^f}(s)}(w), g_{L^f}(w)\right\}, \quad \forall w,\ 0 \le w \le Q,\ s \ge s^f.$$

Labels in the updated buckets which are now dominated due to the addition of the new label, are removed from the corresponding buckets.

Below, we describe the backward extension following the scheme used for the forward one.

Backward extension A backward path $\overline{P} = (i_s, i_{s-1}, \ldots, i_1, i_0 = n')$ is represented by a label L^b associated with the first vertex i_s, where $L^b = (v^b, s^b, V^b, W^b, \{(a_h^b, b_h^b)\}_{h \in H^b})$ is composed of the following information:

- $v^b \in V$: the first vertex visited.
- s^b: the total duration.
- $V^b \subseteq V$: the set of vertices that could be appended before the first vertex v^b.
- W^b: the maximum amount of product required by the delivery customers of the path.
- $\{(a_h^b, b_h^b)\}_{h \in H^b}$: the representation of the piecewise-linear convex nonincreasing function.

A backward label L_1^b with $j = v_1^b$ can be extended to a vertex $i \in V_1^b$, yielding a new label L_2^b based on the following extension functions. We have two cases (see also the proof of Theorem 2):

(i) $i \in N^-$ (pickup customer). The components of label L_2^b are computed as follows:

- $v_2^b = i$.
- $s_2^b = s_1^b + t_{ij}$.
- $V_2^b = V_1^b \setminus \{l : (j,l) \in A, s_2^b + t_{li} + t_{0l} > T\} \setminus \{i\}$.
- $W_2^b = \min\{Q, W_1^b - q_i\}$.
- $$\{(a_h^b, b_h^b)\}_{h\in H_2^b} = \left\{(a_h^b, \quad b_h^b + d_{ij})\right\}_{h\in H_1^{b-}} \bigcup \left\{(a_h^b, b_h^b + (a_h^b - \mu_i)q_i + d_{ij})\right\}_{h\in H_1^{b+}} \bigcup \left\{(\mu_i, \max\left\{z_1^*, \max_{h\in H_1^{b-}}\{(a_h^b - \mu_i)W_1^b + b_h^b\}, \max_{h\in H_1^{b+}}\{b_h^b\}\right\} + d_{ij})\right\},$$

where $H_1^{b-} = \{h \in H_1^b : a_h^b - \mu_i < 0\}$, $H_1^{b+} = \{h \in H_1^b : a_h^b - \mu_i \geq 0\}$ and $z_1^* = \min\max_{h\in H_1^b}\{a_h^b x - b_h^b\}$.

(ii) $i \in N^+$ (delivery customer). Components v_2^b, s_2^b and V_2^b are computed as for the previous case, whereas the remaining components are updated as follows:

- $W_2^b = W_1^b$.
- $$\{(a_h^b, b_h^b)\}_{h\in H_2^b} = \left\{(a_h^b, b_h^b + (a_h^b - \mu_i)q_i + d_{ij})\right\}_{h\in H_1^{b-}} \bigcup \left\{(a_h^b, b_h^b + d_{ij})\right\}_{h\in H_1^{b+}} \bigcup \left\{(-\mu_i, \max\left\{z_1^*, \max_{h\in H_1^{b-}}\{(a_h^b - \mu_i)W_1^b + b_h^b\}\right\} + d_{ij})\right\},$$

where H_1^{b-}, H_1^{b+} and z_1^* are defined as in the previous case.

Backward labels are generated by starting from the initial label $L^b = (n', \{n'\}, 0, N, 0, \{(0,0)\})$, and labels with a capacity consumption larger than or equal to $\lceil T/2 \rceil$ are not generated.

Similar to the forward expansion, the following set-dominance rule is also used to speed up the computation. Let $L_1^b = (v_1^b, s_1^b, V_1^b, W_1^b, \{(a_h^b, b_h^b)\}_{h\in H_1^b})$ and define $\mathscr{L}_1^b$ to be the set of labels $L^b = (v^b, s^b, V^b, W^b, \{(a_h^b, b_h^b)\}_{h\in H^b})$ such that $v^b = v_1^b$, $s^b \geq s_1^b$ and $V_1^b \subseteq V^b$.

Dominance 2 (Backward set-dominance) *Label L_1^b is dominated by the label set $\mathscr{L}_1^b$ if $g_{L_1^b}(w) \geq \overline{g}_{\mathscr{L}_1^b}(w)$, $\forall w$, $0 \leq w \leq Q$.*

We omit the corresponding proof, it being similar to the proof of Dominance 1.

Joining forward and backward states The set of forward and backward labels generated can be combined to form complete routes as follows.

A forward path $P = (i_0 = 0, i_1, \ldots, i_{s-1}, i_s)$ associated with label $L^f = (v^f, s^f, V^f, W^f, \{(a_h^f, b_h^f)\}_{h\in H^f})$ with $i = i_s$ can be combined with a backward path $\overline{P} = (j_l, j_{l-1}, \ldots, j_1, j_0 = n')$ associated with label $L^b = (v^b, s^b, V^b, W^b, \{(a_h^b, b_h^b)\}_{h\in H^b})$ with $j = j_l$ if the total duration is less than or equal to T, i.e., $s^f + t_{ij} + s^b \leq T$.

The reduced cost function of the resulting route $R = (0, i_1, \ldots, i_{s-1}, i, j, j_{l-1}, \ldots, j_1, n')$ can be computed as

$$\overline{c}(R, w) = \max_{h\in H^f}\{a_h^f w + b_h^f\} + d_{ij} + \max_{h\in H^b}\{a_h^b w + b_h^b\},$$

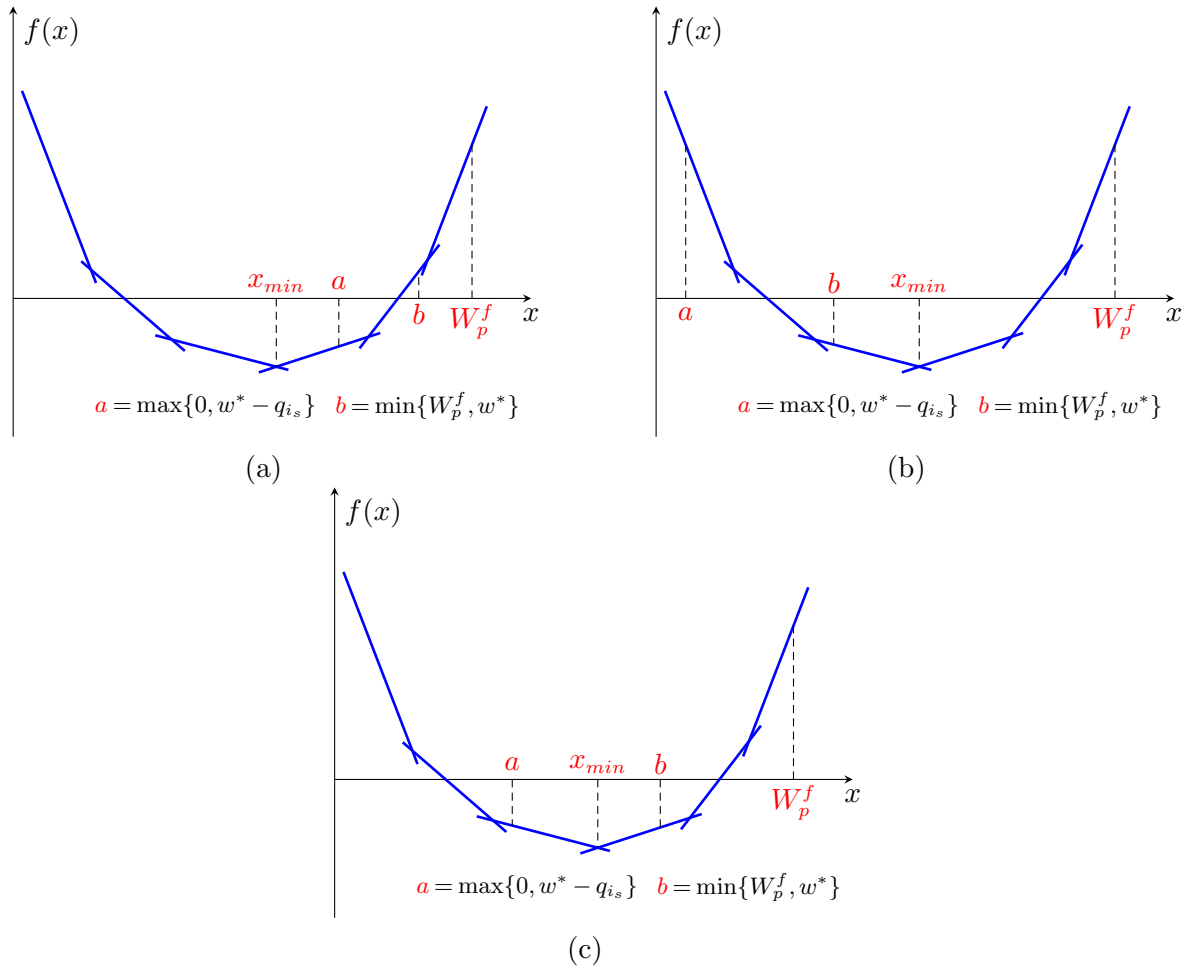

Figure 2 **Computing the optimal value x^*.**

where $0 \leq w \leq \min\{W^f, W^b\}$ is the demand delivered by the partial forward path P associated with label L^f to the backward path $\overline{P}$ associated with label L^b. The value w^* minimizing function $\overline{c}(R, w)$ can therefore be computed as

$$w^* = \underset{0 \leq w \leq \min\{W^f, W^b\}}{\arg\min} \{\overline{c}(R, w)\}.$$

The set of quantities $\{l_{i_1}, \ldots, l_{j_1}\}$ associated with route R and value w^* can be determined by a backtracking procedure starting with the optimal value w^* applied to paths P and $\overline{P}$, as described below.

For the forward path $P = (i_0 = 0, i_1, \ldots, i_{s-1}, i_s)$ and associated label $L^f = (v^f, s^f, V^f, W^f, \{(a_h^f, b_h^f)\}_{h \in H^f})$, let $L_p^f = (v_p^f, s_p^f, V_p^f, W_p^f, \{(a_h^f, b_h^f)\}_{h \in H_p^f})$ be the label immediately preceding label L^f, and let $P' = (i_0 = 0, i_1, \ldots, i_{s-1})$ be the corresponding forward path. The backtracking procedure considers the following two cases:

(i) $i_s \in N^+$ (pickup customer). We have

$$\overline{c}_P^f(w^*) = \min_{\max\{0, w^* - q_{i_s}\} \leq x \leq \min\{W_p^f, w^*\}} \{\overline{c}_{P'}^f(x) - \mu_{i_s}(w^* - x) + d_{i_{s-1} i_s}\},$$

$$= \min_{\max\{0,w^*-q_{i_s}\}\le x\le \min\{W_p^f,w^*\}} \left\{ \max_{h\in H_p^f}\{(a_h^f+\mu_{i_s})x+b_h^f\} - \mu_{i_s}w^* + d_{i_{s-1}i_s} \right\}.$$

Define $f(x) = \max_{h\in H_p^f}\{(a_h^f+\mu_{i_s})x+b_h^f\} - \mu_{i_s}w^* + d_{i_{s-1},i_s}\}$, $H_p^{f+} = \{h\in H_p^f : a_h+\mu_{i_s}\ge 0\}$ and $H_p^{f-} = \{h\in H_p^f : a_h+\mu_{i_s}<0\}$. Let (x_{min}, z_{min}) be the intersection point of the last line in H_p^{f-} and the first line in H_p^{f+}. We set $x_{min} = -\infty$ and $z_{min} = -\infty$ if $H_p^{f-} = \emptyset$ or $H_p^{f+} = \emptyset$. To compute the demand $l_{i_s} = w^* - x^*$ associated with i_s, we have the following three cases, as shown by Figure 2:

(a) If $x_{min} < \max\{0, w^* - q_{i_s}\}$, then $x^* = \max\{0, w^* - q_{i_s}\}$.

(b) If $x_{min} > \min\{W_p^f, w^*\}$, then $x^* = \min\{W_p^f, w^*\}$.

(c) If $\max\{0, w^* - q_{i_s}\} \le x_{min} \le \min\{W_p^f, w^*\}$, then $x^* = x_{min}$.

(ii) $i_s \in N^-$ (delivery customer). We have

$$\begin{aligned}\overline{c}_P^f(w^*) &= \min_{w^*\le x\le \min\{W_p^f, w^*-q_{i_s}\}} \{\overline{c}_{P'}^f(x) - \mu_{i_s}(x-w^*) + d_{i_{s-1}i_s}\},\\ &= \min_{w^*\le x\le \min\{W_p^f, w^*-q_{i_s}\}} \left\{ \max_{h\in H_p^f}\{(a_h^f-\mu_{i_s})x+b_h^f\} + \mu_{i_s}w^* + d_{i_{s-1}i_s} \right\}.\end{aligned}$$

Let $H_p^{f+} = \{h\in H_p^f : a_h - \mu_{i_s}\ge 0\}$ and $H_p^{f-} = \{h\in H_p^f : a_h-\mu_{i_s}<0\}$. To compute the demand $l_{i_s} = x^* - w^*$, we have the following three cases:

(a) If $x_{min} < w^*$, then $x^* = w^*$.

(b) If $x_{min} > \min\{W_p^f, w^* - q_{i_s}\}$, then $x^* = \min\{W_p^f, w^* - q_{i_s}\}$.

(c) If $w^* \le x_{min} \le \min\{W_p^f, w^* - q_{i_s}\}$, then $x^* = x_{min}$.

For the backward path $\overline{P} = (j_l, j_{l-1}, \ldots, j_1, j_0 = n')$ and associated label $L^b = (v^b, s^b, V^b, W^b, \{(a_h^b, b_h^b)\}_{h\in H^b})$, let $L_p^b = (v_p^b, s_p^b, V_p^b, W_p^b, \{(a_h^b, b_h^b)\}_{h\in H_p^b})$ be the label immediately preceding label L^b, and let $\overline{P}' = (j_{l-1}, \ldots, j_1, j_0 = n')$ be the corresponding backward path. The backtracking procedure considers the following two cases:

(i) $j_l \in N^+$ (pickup customer). We have

$$\begin{aligned}\overline{c}_{\overline{P}}^b(w^*) &= \min_{w^*\le x\le \min\{W_p^b, w^*+q_{j_l}\}} \{\overline{c}_{P'}^b(x) - \mu_{j_l}(x-w^*) + d_{j_l j_{l-1}}\},\\ &= \min_{w^*\le x\le \min\{W_p^b, w^*+q_{j_l}\}} \left\{ \max_{h\in H_p^b}\{(a_h^b-\mu_{j_l})x+b_h^b\} + \mu_{j_l}w^* + d_{j_l j_{l-1}} \right\}.\end{aligned}$$

Let $H_p^{b+} = \{h\in H_p^b : a_h - \mu_{j_l}\ge 0\}$ and $H_p^{b-} = \{h\in H_p^b : a_h - \mu_{j_l} < 0\}$. We have the following three cases:

(a) If $x_{min} < w^*$, then $x^* = w^*$.

(b) If $x_{min} > \min\{W_p^b, w^* + q_{j_l}\}$, then $x^* = \min\{W_p^b, w^* + q_{j_l}\}$.

(c) If $w^* \le x_{min} \le \min\{W_p^b, w^* + q_{j_l}\}$, then $x^* = x_{min}$.

The demand l_{i_l} is set equal to $x^* - w^*$.

(ii) $j_l \in N^-$ (delivery customer). We have

$$\overline{c}^b_{\overline{P}}(w^*) = \min_{\max\{0,w^*+q_{j_l}\} \le x \le \min\{W^b_p, w^*\}} \{\overline{c}^b_{P'}(x) - \mu_{j_l}(w^* - x) + d_{j_l j_{l-1}}\},$$

$$= \min_{\max\{0,w^*+q_{j_l}\} \le x \le \min\{W^b_p, w^*\}} \left\{ \max_{h \in H^b_p} \{(a^b_h + \mu_{j_l})x + b^b_h\} - \mu_{j_l} w^* + d_{j_l j_{l-1}} \right\}.$$

Let $H^{b+}_p = \{h \in H^b_p : a_h + \mu_{j_l} \ge 0\}$ and $H^{b-}_p = \{h \in H^b_p : a_h + \mu_{j_l} < 0\}$. We have the following three cases:

(a) If $x_{min} < \max\{0, w^* + q_{j_l}\}$, then $x^* = \max\{0, w^* + q_{j_l}\}$.

(b) If $x_{min} > \min\{W^b_p, w^*\}$, then $x^* = \min\{W^b_p, w^*\}$.

(c) if $\max\{0, w^* + q_{j_l}\} \le x_{min} \le \min\{W^b_p, w^*\}$, then $x^* = x_{min}$.

The demand l_{i_l} is set equal to $w^* - x^*$.

The minimum reduced cost among all complete routes is the optimal solution value of the pricing subproblem.

5. An exact algorithm for the SPDVRP

In this section, we describe an exact algorithm for the SPDVRP based on a BPC algorithm. The reader is referred to Costa et al. (2019) and Pecin et al. (2017) for a recent review about the main methodological and modeling contributions made over the years on BPC algorithms for routing problems, and for an exact BPC algorithm that includes state-of-the-art features for the VRPTW, respectively.

5.1. Column-and-row generation

At a given node of the branch-and-bound search tree, the LP-relaxation corresponds to formulation $\overline{LPB}$, augmented by the applicable branching decisions and the following valid inequalities.

Given a route $r \in \mathcal{R}$ and a pattern index $p \in \mathcal{P}_r$, coefficient β_{ir} is a nonnegative positive integer representing the number of times that route R_r visits vertex i (we assume $\beta_{0r} = \beta_{n'r} = 1$), and we let set $\mathscr{S}$ be defined as $\mathscr{S} \subseteq \{S : S \subseteq N, |S| \ge 2\}$.

Formulation $\overline{LPB}$ can be strengthened with the following valid inequalities called Capacity Cuts (CCs) based on similar inequalities designed for the CVRP (see, for example, Baldacci et al. 2012), where the set of customers comprises only delivery customers and split deliveries are not allowed.

The CCs for the CVRP assume that, given a subset $S \in \mathscr{S}$ of customers, one can estimate a lower bound $k(S)$ on the number of vehicles required to serve all customers in S. In this case, at least $k(S)$ paths must enter S in a feasible solution. Let $(x_{ij})_{(i,j \in A)}$ be arc flow variables. The inequality for the CVRP is as follows:

$$\sum_{i \in V \setminus S, j \in S} x_{ij} \ge k(S),$$

where the left-hand-side of the inequality is the total flow entering S. Given a solution θ_{pr} of formulation $\overline{LPB}$, the arc flow variables can be computed using the following expression:

$$x_{ij} = \sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \gamma_{ijr}\theta_{pr},$$

and therefore by setting $k(S) = \max\left\{1, \left\lceil \frac{|\sum_{i\in S} q_i|}{Q} \right\rceil\right\}$ we derive the following valid inequalities:

$$\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \rho_r(S)\theta_{pr} \geq k(S),\ \forall S \in \mathscr{S}, \tag{4}$$

where $\rho_r(S) = \sum_{(i,j)\in A: i\in S, j\in V\setminus S} \gamma_{ijr}$. To separate CCs, we use partial enumeration and extended shrinking separation heuristics from the literature (Desaulniers 2010, Archetti, Bouchard and Desaulniers 2011, Luo et al. 2021).

The LP-relaxation is solved in a column-and-row generation fashion where, at each iteration, the primal simplex algorithm is used to solve the RMP and to provide a primal and a dual solution. Then the pricing subproblem (*column generation*) is solved using the label-setting algorithm in order to find negative reduced cost columns or variables. If no negative reduced cost columns can be found, the current primal solution is optimal for the master problem. Otherwise, one or several negative reduced cost columns are added to the RMP before beginning a new iteration. If the current primal solution is optimal, valid inequalities (4) are separated in a cutting plane fashion (*row generation*). The cutting plane algorithm terminates when no additional valid inequalities are identified by the separation algorithms, and a new iteration of the column-and-row generation is executed. The lower bound computation stops when both the column and the row generation algorithms terminate without having found new columns/rows to be added to the RMP.

In the computational experiments reported in Section 6, at each column generation iteration we identify at most 50 columns with negative reduced cost. In the following, we describe how the RMP is initialized at the root node of the enumeration tree, the branching scheme adopted in the algorithm, how the CCs and the branching rules are handled in the pricing algorithm, and the computation of primal bounds.

5.2. Initializing the RMP

The set of CCs $\mathscr{S}$ is initialized as the empty set, and the initial columns forming the RMP are computed by means of a constructive algorithm of which details are given in Algorithm 1.

In the algorithm, S_P and S_D denote the sets of pickup and delivery vertices, respectively, $\overline{q}_i$ represents the residual demand of customer $i \in N$, and ld_i is the load collected from or delivered to customer $i \in N$. In addition, δ_i represents the load that has been collected from $i \in S_P$ and has not been delivered yet. The final set of routes, with the corresponding demand patterns built by the algorithm and used to initialize the RMP, is denoted by $\mathcal{B}$.

Algorithm 1 Initialization of the RMP.

Set $\mathcal{B} \leftarrow \emptyset$, $S_P \leftarrow N^+$, $S_D \leftarrow N^-$, $\overline{q}_i \leftarrow q_i$, $\forall\ i \in N$, and $\mathcal{L} \leftarrow \emptyset$;
while $S_P \neq \emptyset$ **do**
  Select $i \in S_P$ with the largest $\overline{q}_i$ and define the single-customer route $R \leftarrow (0, i, n')$;
  $ld_i \leftarrow \min\{Q, \overline{q}_i\}$, $\delta_i \leftarrow ld_i$, $\overline{q}_i \leftarrow \overline{q}_i - ld_i$;
  if $\overline{q}_i = 0$ **then**
    $S_P \leftarrow S_P \setminus \{i\}$;
  end if
  while $\delta_i > 0$ **do**
    Select $i \in S_D$ as the nearest customers to the last customer of route R;
    $ld_j \leftarrow \max\{-\delta_i, \overline{q}_j\}$, $\delta_i \leftarrow \delta_i + ld_j$, $\overline{q}_j \leftarrow \overline{q}_j - ld_j$;
    if $\overline{q}_j = 0$ **then**
      $S_D \leftarrow S_D \setminus \{j\}$;
    end if
    Append j after the last customer of route R;
  end while
  $\mathcal{B} \leftarrow \mathcal{B} \cup \{R\}$;
end while

At each main iteration of the algorithm, the pickup customer having the maximum residual demand $\overline{q}_i$ is selected to initialize a new route with the corresponding demand-pattern. The route is then expanded with delivery customers until the load that has been collected from i (and has not been delivered yet) is fully distributed. The algorithm terminates whenever the total demand of the pickup customers has been delivered and, since $\sum_{i \in V} q_i = 0$, also the total demand of the delivery customers has been satisfied.

The set of routes and corresponding demand patterns computed by Algorithm 1 do not necessarily define a feasible solution of the RMP, since constraint (1c) on the maximum number of vehicles available can be violated. To find an initial basic feasible solution of the RMP, we add a single artificial variable with a large positive cost and with an associated column consisting of all ones for entries in correspondence of constraints (1b) and (1c). This artificial variable ensure that a feasible solution to the LP-relaxation exists.

5.3. Branching strategy

The branching strategy adopted is based on the strategy described by Desaulniers (2010) for the SDVRPTW, and uses four types of branching decisions defined as follows.

Let $\overline{\theta}$ be an optimal solution of the RMP. The following four types of branching decisions are considered in sequence, and if a branching decision cannot be applied, the next decision in the

sequence is considered in order to perform branching. For any type of decision made, two child nodes are created by performing dichotomic branching.

(1) *Number of vehicles used.* Define $\overline{k} = \sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \overline{\theta}_{pr}$. If $\overline{k}$ is fractional, we branch on the value of $\overline{k}$, that is, on a first branch we impose $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \overline{\theta}_{pr} \geq \lceil \overline{k} \rceil$, whereas on a second branch we impose $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \overline{\theta}_{pr} \leq \lfloor \overline{k} \rfloor$.

(2) *Number of vehicles visiting each customer.* Define $\overline{\sigma}_i = \sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \beta_{ir}\overline{\theta}_{pr}$, $\forall i \in N$. If there exists at least one customer $i \in N$ such that $\overline{\sigma}_i$ is fractional, then we select the customer i^* for which the fractional part of $\overline{\sigma}_i$ is the closest to 0.5. On a first branch, we impose $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \beta_{i^*r}\overline{\theta}_{pr} \geq \lceil \overline{\sigma}_{i^*} \rceil$, whereas on a second branch, we impose $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \beta_{i^*r}\overline{\theta}_{pr} \leq \lfloor \overline{\sigma}_{i^*} \rfloor$.

(3) *Branching on arcs.* Let $\overline{\omega}_{ij} = \sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \gamma_{ijr}\overline{\theta}_{pr}$. We consider the following two cases, which are considered in sequence.

   (i) Let $A^+ = \{(i,j) : (i,j) \in A, i,j \in N^+\}$ and $A^- = \{(i,j) : (i,j) \in A, i,j \in N^-\}$. It can be shown that, based on a similar property arising for the SDVRP (Dror and Trudeau 1989), in any optimal SPDVRP solution, any arc $(i,j) \in A^+ \cup A^-$ can be used at most once. Hence, if there exists at least one arc $(i,j) \in A^+ \cup A^-$ such that $\overline{\omega}_{ij}$ is fractional, then we select the arc (i^*,j^*) for which the fractional part of (i,j) is the closest to 0.5. On a first branch, we impose $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \gamma_{i^*j^*r}\overline{\theta}_{pr} = 0$, and on a second branch we impose $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \gamma_{i^*j^*r}\overline{\theta}_{pr} = 1$.

   (ii) If there exists at least one arc $(i,j) \in A \setminus (A^+ \cup A^-)$ such that $\overline{\omega}_{ij}$ is fractional, we select the arc (i^*,j^*) as before, and we impose the two branches $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \gamma_{i^*j^*r}\overline{\theta}_{pr} \geq \lceil \overline{\omega}_{i^*j^*} \rceil$ and $\sum_{r\in\mathcal{R}} \sum_{p\in\mathcal{P}_r} \gamma_{i^*j^*r}\overline{\theta}_{pr} \leq \lfloor \overline{\omega}_{i^*j^*} \rfloor$.

(4) *Branching on two consecutive arcs.* Let $\overline{\omega}_{ijl} = \sum_{r\in\mathcal{R}_{ijl}} \sum_{p\in\mathcal{P}_r} \overline{\theta}_{pr}$, where $\mathcal{R}_{ijl}$ denotes the index set of routes containing arc (j,l) immediately after arc (i,j). If there exists two arcs (i,j) and (j,l) such that $\overline{\omega}_{ijl}$ is fractional, and either (i,j) or (j,l) belong to $A^+ \cup A^-$, we select (i^*,j^*) and (j^*,l^*) such that $\overline{\omega}_{i^*j^*l^*}$ is the closest to 0.5, and on a first branch we impose $\sum_{r\in\mathcal{R}_{i^*j^*l^*}} \sum_{p\in\mathcal{P}_r} \overline{\theta}_{pr} = 0$, that is, the use of arc (j^*,l^*) immediately after arc (i^*,j^*) is forbidden, and on a second branch we impose $\sum_{r\in\mathcal{R}_{i^*j^*l}} \sum_{p\in\mathcal{P}_r} \overline{\theta}_{pr} = 0$, $\forall (j^*,l)$, $l \neq l^*$, $\forall (j^*,l) \in A$, i.e., if arc (i^*,j^*) is used, also arc (j^*,l^*) must also be used.

The above branching decisions are not sufficient to fulfill the integrality requirements of formulation $\overline{PB}$, as shown by the example reported in the e-compation to this paper (see §EC.2). If none of the decisions (1)-(4) can be applied at a given node of the enumeration tree, we solve the SPDVRP subproblem associated with the node by means of a BC algorithm based on a three-index vehicle flow formulation (see §EC.3) strengthened by the branching constraints associated with the node. In the computational experiments reported on in Section 6, the above four branching rules

were sufficient to impose the integrality requirements, and the BC algorithm was therefore never applied.

To choose a node-selection rule, we performed some preliminary experiments with different rules and, based on these results, we adopted the *best-first strategy*.

5.4. Handling the CCs and the branching rules in the pricing algorithm

According to the classification of Poggi and Uchoa (2003), the CCs are called *robust cuts* because they do not increase the complexity of the pricing problem. Indeed, let $\nu_S \geq 0, \forall S \in \mathscr{S}$, be the dual variables associated with constraints (4). The dual variables of the CCs can be easily considered in the label-setting algorithm by defining the modified arc costs $\overline{d}_{ij}$ as follows

$$\overline{d}_{ij} = d_{ij} - \sum_{S \in \mathscr{S}_{ij}} \nu_S, \; \forall (i,j) \in A,$$

where $\mathscr{S}_{ij} = \{S \in \mathscr{S} : (i,j) \in A, i \in S, j \in V \setminus S\}$, and by substituting arc costs d_{ij} with the modified costs $\overline{d}_{ij}$ in the label-setting algorithm described in Section 4.2.

Regarding the branching rules, branching rules (1)-(3) preserve the structure of the pricing problem, that is, the corresponding decisions can be easily handled in the label-setting algorithm, whereas branching rule (4) requires modifying the label-setting algorithm to impose suitable extensions based on the corresponding branching constraints. Indeed, regarding rule (4), the procedure used to check the dominance rules described in §4.2 must be modified to ensure that a label generated as a result of a branching constraint on arc (i,j) can only dominate labels generated via label extensions along arc (i,j).

Let Π_i be the set of vertices such that arcs (j,i) for $j \in \Pi_i$ have been selected as arcs in branching decisions (4). The labels ending at customer i are decomposed in buckets $\mathscr{F}_i(j,s)$ for $\forall j \in \Pi_i$ and an additional bucket $\mathscr{F}_i(-,s)$, which are organized in a bucket tree structure. A label $L^f = (v^f, s^f, \{(a_h^f, b_h^f)\}_{h \in Hf})$ with $v^f = i$ and predecessor vertex j, is checked for dominance (see §4.2) in bucket $\mathscr{F}_i(j, s^f)$ if $j \in \Pi_i$, otherwise bucket $\mathscr{F}_i(-, s^f)$ is selected. Figure 3 gives an example of a bucket tree for labels ending at customer $i = 3$ with and $\Pi_3 = \{2,4\}$, where $s_1^0 < s_2^0 < s_3^0 < s_4^0$, $s_1^1 < s_2^1 < s_3^1$ and $s_1^2 < s_2^2$.

5.5. Primal heuristic

As shown by previous works (Joncour et al. 2010, Sadykov et al. 2019, Sadykov and Vanderbeck 2013, Wei et al. 2020), column generation based heuristics can help to improve the performance significantly. More specifically, Joncour et al. (2010) described a generic solution framework based on a depth-first heuristic search in the BP tree, named *diving heuristic*. In the framework proposed by Joncour et al. (2010), the solution obtained through the initial depth-first exploration of the tree

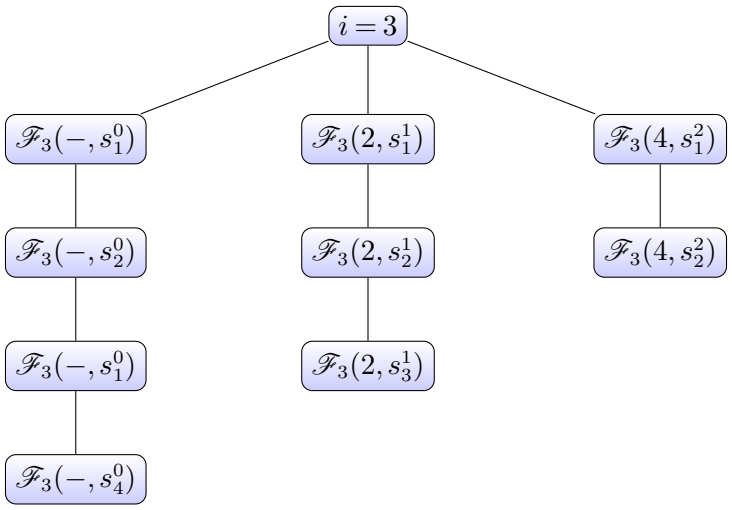

Figure 3 Example of a bucket tree

is considered as a reference incumbent solution and it is further explored using limited backtracking as a diversification mechanism and limited discrepancy search. For the SPDVRP, we designed a diving heuristic based on a BP tree, where we adopted the branching and search strategies described below.

At each BP node, the master is solved by column generation where the separation algorithm of the valid inequalities (4) is disabled. The BP enumeration is driven by a binary branching by first selecting a variable θ_{pr} whose fractional value θ^*_{pr} is as close as possible to 0.5, and ties are broken by selecting the variable having the minimum cost. Then two branches are generated by adding constraints $\theta_{pr} \geq \lceil \theta^*_{pr} \rceil$ (*left branch*) or $\theta_{pr} \leq \lfloor \theta^*_{pr} \rfloor$ (*right branch*) to the RMP. In the two branches, the demand pattern α_{ipr} associated with variable θ_{pr} is eventually modified to ensure that for each vertex i of route R the demand $|\alpha_{ipr}|\, v$ is less than or equal to $|q_i|$, where v is either equal to $\lceil \theta^*_{pr} \rceil$ or $\lfloor \theta^*_{pr} \rfloor$, thus ensuring the feasibility of the resulting RMP. The pricing algorithm is not modified to include the branching constraints, and if a column composing the RMP is regenerated during the solution of the pricing problem, the column is not added to the RMP.

The tree is explored using a modified depth-first strategy where we restrict the search to a limited number of nodes. More specifically, suppose $(n_1, ..., n_i)$ is a node sequence where n_1 is the root node and node n_{j-1} is the father of node n_j, $j = 2, \ldots, i$. Thus, if n_j is the left child node of n_{j-1}, $j = 2, \ldots, i$, then the next node explored by the algorithm, i.e., node n_{i+1}, is the right child of n_i; otherwise, n_{i+1} can only be the left child node of n_i and the right child is discarded. Therefore the search is restricted by choosing at most one right child node in the path from the root node to any visited node.

Based on preliminary experiments, we found it computationally convenient to use the primal heuristic only at the root node of the BPC algorithm.

6. Computational experiments

This section reports on the computational results of the exact algorithm described in Section 5, hereafter called EXM.

Method EXM was coded in Java, and all experiments were conducted on a personal computer equipped with an Intel(R) Core(TM) i7-6700 3.40 GHz CPU and 32 GB of RAM, running under Windows 8.1 operating system. The CPLEX 12.6.4 callable library (IBM CPLEX 2019) was used as the linear programming solver.

The next section reports on the description of the instances used in our numerical experiments whereas Section 6.2 gives a summary of the results obtained. We conclude this section (see §6.3) with an analysis of the performance of the pricing algorithm of EXM. Additional details are given in the e-companion to this paper (see §EC.5).

6.1. Benchmark instances

We considered three classes of test instances, called S_A, S_B and S_C, respectively. In classes S_A and S_C the route duration constraints are used to impose an upper limit on the number of stops on each route whereas class S_C considers general route duration constraints.

Class S_A has been obtained from the literature and contains the set of MVBP instances used by Casazza et al. (2019) and Casazza et al. (2021), which were derived from the instances proposed by Chemla et al. (2013). In these instances, the customers are randomly located in the square $[-500, 500]$ and have integer demands randomly chosen in the interval $[-10, 10]$; the depot is located at point (0, 0). The travel costs are computed as the Euclidean distances, rounded to the closest integer numbers. Casazza et al. (2019) and Casazza et al. (2021) considered instances with the number of customers n in $\{10, 20, 30\}$ and with $T = 10$ (a maximum number of stops on each route is imposed). In our experiments, we also considered instances with $n = 40$ and $T = 15$. The vehicle capacity Q and the number of vehicles $|K|$ are chosen from sets $\{10, 20\}$ and $\{5, 8\}$, respectively. To model a maximum number of stops b in the SPDVRP we set $T = b$ and $t_{ij} = 1$ for all $(i, j) \in A$. A total of 200 instances have been used, with names of the form <nXY>, where X is the number of stations or customers and Y a letter identifying the type of network used in deriving the instance.

Class S_B of instances corresponds to the set of instances used by Casazza et al. (2021) and is also derived from the instances of Chemla et al. (2013) by properly defining the travel time matrices $[t_{ij}]$. More specifically, instances in the dataset of Chemla et al. (2013) were considered in lexicographical order, and for each instance, the travelling times t_{ij} were set as the Euclidean distance between vertices i and j in the subsequent instance according to the lexicographical order, thus obtaining unrelated travel times and costs. The maximum duration T was set equal to either 3600 or 7200. Casazza et al. (2021) considered instances with up to $n = 30$ customers and we also

considered instances involving $n = 40$ customers. A total of 100 instances were considered, and this set of instances uses the same naming convention as set S_A.

Class S_C was derived from the set of real-world instances considered by Dell'Amico et al. (2014) and based on different datasets. More specifically, the authors collected data from a 7-month period concerning the bike-sharing system of the city of Reggio Emilia, in Northern Italy, this data being drawn from Capital Bikeshare (see `https://www.capitalbikeshare.com`) and as well as from Divvy, located in Chicago, USA (see `https://www.divvybikes.com`). All the instances are available at `http://www.or.unimore.it/site/home/online-resources.html`. From this set, we considered a total of 47 instances, with the number of customers ranging in $[13, 55]$ and the vehicle capacity in $[10, 30]$. For these instances, we imposed a maximum number of stops T equal to 10 or 15, and the number of available vehicles was set equal to $|K| = 8$ for all instances. A total of 94 instances were used, with names in the form of <iXQ>, where i is the instance index, X the name of the corresponding city, and Q the vehicle capacity.

For the convenience of readers, we summarize the three classes of instances used and corresponding results solved by EXM at `https://github.com/huster-ljl/SPDVRP`.

6.2. Results

This section summarizes the results obtained by EXM and compares its performances with the results reported in Casazza et al. (2019) and Casazza et al. (2021) on classes S_A and S_B. The results of Casazza et al. (2019) and Casazza et al. (2021) were obtained on a PC equipped with an Intel Core i76700K CPU clocked at 4.00 GHz, thus having similar performance to the machine used for our experiments, and a time limit of 3 hours of computation was imposed in their experiments. In the following, we denote by C18 and C21 the algorithms of Casazza et al. (2019) and Casazza et al. (2021), respectively. All the computing times reported in this section are expressed in seconds, and a time limit of 3 hours was also imposed to each execution of EXM.

Table 2 summarizes the results obtained by the algorithms on the three classes of instances. For each class of instances and for each subgroup of instances in the set, the table gives the range of the number of customers ("n"), the number of instances in the group ("ni"), the values of parameters Q and T, and for each algorithm the number of instances solved to optimality ("*opt*"), the size of the largest instance solved to optimality in terms of number of customers ("$\hat{n}$") and the average computing time in seconds ("t") computed over all instances solved to optimality. In addition, Figure 4 gives an overview of the percentages of the number of instances solved to optimality by the different algorithms grouped by number of customers (horizontal axis) and by considering, for each algorithm, the set of instances addressed in the corresponding experiments. As already mentioned, algorithms C18 and C21 were tested on instances with up to 30 customers.

Table 2 Summary of the results on the three classes of instances

					C18			C21			EXM		
Class	n	ni	Q	T	opt	$\hat{n}$	t	opt	$\hat{n}$	t	opt	$\hat{n}$	t
S_A	10-40	50	10	10	25	20	1000.9	24	20	265.7	35	30	664.3
		50	10	15	-	-	-	-	-	-	32	30	265.5
		50	20	10	24	20	2483.9	27	20	1016.0	34	40	679.9
		50	20	15	-	-	-	-	-	-	33	40	368.5
S_B	10-40	50	10	3600	-	-	-	21	20	319.3	33	30	531.9
		50	10	7200	-	-	-	21	20	125.6	31	30	177.2
S_C	13-55	47	10-30	10	-	-	-	-	-	-	34	28	695.2
		47	10-30	15	-	-	-	-	-	-	32	41	1001.1

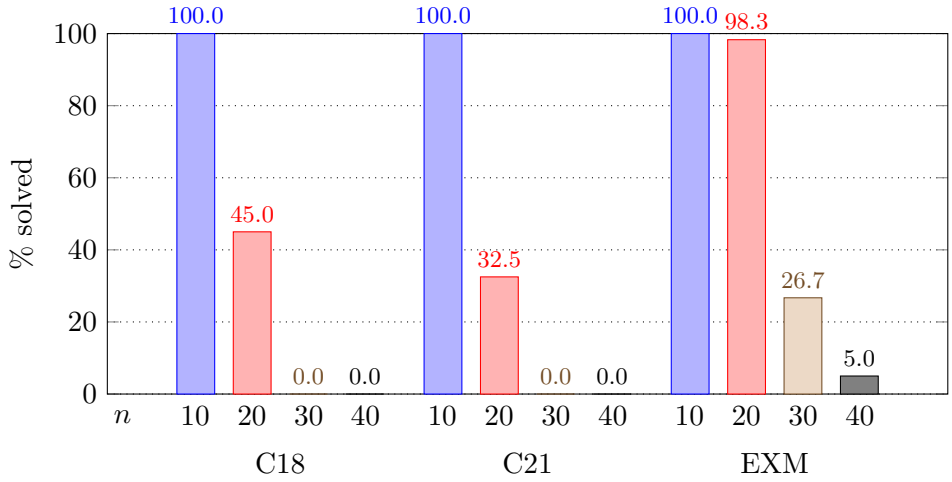

Figure 4 Percentages of the instances solved to optimality by the different algorithms on classes S_A and S_B

The table shows that about 70% of the instances of each class were solved to optimality by EXM within the imposed time limit. Overall, EXM solved to optimality instances with up to 41 customers. As to the set of instances considered by C18 and C21, EXM is capable of solving far more instances to optimality. A comparison of the results given in Casazza et al. (2021) about C18 and C21 with the detailed results reported on in the e-companion to this paper shows that all instances solved to optimality by C18 and C21 are also solved by EXM. It is worth noting that there are no dominance relations between C18 and C21, that is, there are instances solved by C18 within the imposed time limit that cannot be solved by C21, and viceversa.

Regarding classes S_B and S_C, Figure 4 clearly shows that EXM is capable of solving larger size instances than both C18 and C21. In particular, the detailed results show that EXM solved all but one instance involving $n = 20$ customers. Several instances with up to 40 customers were solved by EXM, thus doubling the size of the instances that can be solved.

Table 3 reports average results about the lower and upper bounds computed by EXM at the root node of the enumeration tree. These data are not reported in Casazza et al. (2019) and Casazza et al. (2021). The table shows the average percentage deviations of the upper bound ("$\%UB$") computed by the primal heuristic and of the lower bound obtained at the root node ("$\%LB_r$") by the column-and-row generation procedure. The table also reports the corresponding average computing times in seconds ("t_{UB}" and "t_r"). The averages are computed separately for the instances solved to optimality ("Solved to optimality"), and for the instances not solved to optimality ("Not solved to optimality"). For each instance, the percentage deviation is computed as $100.0 \times x/z^*$ where x is the target value and z^* is the cost of the best solution found by EXM.

For the instances solved to optimality, EXM computes tight lower and upper bounds for the real-world instances of class S_C, whereas the bounds computed on classes S_A and S_B show that these instances are more difficult for EXM. We observed that the computation of the upper bound by the primal heuristic helps in improving the performance of EXM and, as shown by the table, the upper bounds computed can be further improved by the BPC.

Based on the results obtained, the two main parameter data that affect the computational complexity of EXM are clearly the number of customer n, as shown by Figure 4 and the detailed results, and the maximum duration T. Indeed, Table 2 shows that for each class and for fixed values of Q, the higher the value of T, the lesser is the number of instances solved to optimality. Both values Q and T affect the computational complexity of the pricing problem, but as also shown by Table 2 and by the analyses reported on the parameter Q in the next section, for fixed values of T and varying values of Q, the pricing algorithm is particularly stable. In the e-companion to this paper (see §EC.4.1), we give an overview of the use of the different branching rules adopted in EXM for classes S_A, S_B and S_C, respectively. Finally, we also analyze the structure of the solutions obtained in the e-companion (see §EC.4.2), with the aim of learning what motivates the differences in our results between the real-world instances of class S_C and classes S_A and S_B.

Finally, the detailed results reported in the e-companion to this paper (see §EC.5)show that the CCs are particularly effective in strengthening the lower bound obtained from the LP-relaxation of formulation $\overline{PB}$.

6.3. Effectiveness of the pricing algorithm

In this section, we report a comparison between the pricing algorithm of EXM and our implementation of the pricing algorithm described by Casazza et al. (2019). The pricing algorithm of Casazza et al. (2019) solves the pricing problem associated with formulation PB and we tailored it to solve the pricing problem associated with relaxation $\overline{PB}$ (see §3.2).

For the comparison, we considered set S_A of instances with $Q = 10$, $T = 10$ and the following two additional sets of instances, for a total of 150 instances (50×3): (i) Set S_A-3 with $Q = 3\,Q$,

Table 3 Summary of lower and upper bounds of EXM

				Solved to optimality				Not solved to optimality			
Class	n	Q	T	$\%UB$	t_{UB}	$\%LB_r$	t_r	$\%UB$	t_{UB}	$\%LB_r$	t_r
S_A	10-40	10	10	112.0	2.7	95.9	1.1	100.0	13.6	77.1	3.5
		10	15	120.7	3.0	98.5	1.3	104.1	43.5	78.5	13.6
		20	10	116.9	2.6	92.5	0.9	101.6	11.2	73.8	2.6
		20	15	125.9	2.8	97.1	1.0	113.4	23.6	82.2	6.0
S_B	10-40	10	3600	112.4	3.7	96.0	1.2	101.4	49.6	79.2	8.6
		10	7200	115.2	7.1	98.6	3.3	102.8	129.9	82.8	41.7
S_C	13-55	10-30	10	105.4	2.7	97.7	1.0	100.5	398.2	91.2	88.7
		10-30	15	104.3	4.8	98.8	1.6	101.8	208.8	92.5	41.7

Table 4 Summary of the performance of alternative pricing algorithms: percentage ratios of the results obtained by the pricing algorithm of EXM with those based on the algorithm of Casazza et al. (2019).

	t_{LP}	t_{PA}	fw	fwd	bw	bwd
S_A	29.66	29.45	6.27	2.65	11.82	4.81
S_A-3	2.68	2.56	0.63	0.09	1.46	0.21
S_A-6	0.21	0.20	0.09	0.01	0.18	0.01

$q_i = 3\,q'_i$ and $T = 10$, and (ii) Set S_B-6 with $Q = 6\,Q$, $q_i = 6\,q'_i$ and $T = 10$, where customer demands q'_i are the original demands of set S_A. The two additional instance sets were generated with the aim of evaluating the impact on the pricing algorithm for increasing values of the vehicle capacity and the customer demands.

Table 4 summarizes the results obtained on the three sets of instances. The corresponding detailed results are reported in the e-companion to this paper (see §EC.6). For each algorithm, we computed the following average results over the different sets of instances: the time in seconds spent by the LP solver ("t_{LP}"), the time in seconds spent by the pricing algorithm ("t_{PA}"), the number of forward labels generated ("fw"), the number of forward labels dominated, ("fwd"), the number of backward labels generated ("bw") and the number of backward labels dominated ("bwd"). Table 4 shows the percentage ratios of the average values t_{LP}, t_{PA}, fw, fwd, bw and bwd obtained by the pricing algorithm of EXM with those based on the algorithm of Casazza et al. (2019).

The table shows that on set S_A (see column t_{PA}) our pricing algorithm is about three times faster than the algorithm based on Casazza et al. (2019). For increasing values of Q associated with sets S_A-3 and S_A-6, our pricing algorithm achieves much more significant speed factors. Indeed, Table 4 and the detailed results show that over the three sets of instances the number of labels generated by our pricing algorithm remains quite stable whereas the alternative pricing algorithm

shows an exponential explosion of the labels generated, as also shown by the ratios over the values fw and bw.

Under the assumption that our implementation of the algorithm of Casazza et al. (2019) achieves similar performance to the implementation of Casazza et al. (2019), the results obtained clearly show that the effectiveness of EXM in solving larger instances (see Table 2) of the SPDVRP can be mainly attributed to the effectiveness of the pricing algorithm based on reduced cost functions.

In conclusion, due to the use of the reduced cost functions, our pricing algorithm requires a reduced number of labels at the cost of more complex dominance rules. However, our results show that the benefits brought about by the label reduction are greater than the loss caused by the higher complexity of our dominance rules.

7. Conclusions and future research

We have presented an exact algorithm for the single commodity split pickup and split delivery vehicle routing problem. The algorithm relies on a novel label-setting (pricing) algorithm, embedding reduced cost functions in the label definition, and being enhanced by new set-dominance rules. The pricing algorithm is used in a branch-price-and-cut algorithm together with additional components such as valid inequalities and a primal heuristic.

Our new exact algorithm was tested on benchmark sets from the literature and on instances derived from real-world bike-sharing problems, involving up to 55 customers. The computational results show that about 70% of the instances, involving up to 41 customers, were effectively solved to optimality by the exact algorithm, thus doubling the size of the instances that can be solved and significantly outperforming the state-of-the-art exact algorithms.

Our algorithm can be easily adapted to deal with other routing constraints, such as time windows constraints, and to solve rich variants involving, for example, a fleet of heterogeneous vehicles. However, the class of split-demand one-commodity pickup and delivery routing problems also features important variants, such as allowing preemption operations. Our future work goes in the direction of extending the current solution approach to incorporate additional important features of this class of problems.

Acknowledgment

The authors would like to thank the anonymous reviewers and associate editor for their helpful suggestions and very thorough review of the paper. The authors also thank Marco Casazza for providing the instances and the corresponding details used in Casazza et al. (2021).

Dr. Hu Qin is the corresponding author in this paper. This research was supported by the National Natural Science Foundation of China (No. 72222011, 71971090, 71821001). This work was also partially supported by the Research Grants Council of Hong Kong SAR, China (No. 15221619).

References

Archetti, C., Bianchessi, N. and Speranza, M. G. (2011), A column generation approach for the split delivery vehicle routing problem, *Networks* **58**(4), 241 – 254.

Archetti, C., Bianchessi, N. and Speranza, M. G. (2014), Branch-and-cut algorithms for the split delivery vehicle routing problem, *European Journal of Operational Research* **238**(3), 685 – 698.

Archetti, C., Bouchard, M. and Desaulniers, G. (2011), Enhanced branch and price and cut for vehicle routing with split deliveries and time windows, *Transportation Science* **45**(3), 285 – 298.

Archetti, C. and Speranza, M. G. (2008), The split delivery vehicle routing problem: A survey, *in* B. L. Golden, S. Raghavan and E. A. Wasil, eds, 'The Vehicle Routing Problem: Latest Advances and New Challenges', Springer, New York, pp. 103 – 122.

Archetti, C. and Speranza, M. G. (2012), Vehicle routing problems with split deliveries, *International Transactions in Operational Research* **19**(1-2), 3 – 22.

Baldacci, R., Christofides, N. and Mingozzi, A. (2008), An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts, *Mathematical Programming* **115**, 351–385.

Baldacci, R., Mingozzi, A. and Roberti, R. (2012), Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints, *European Journal of Operational Research* **218**(1), 1–6.

Balinski, M. L. and Quandt, R. E. (1964), On an integer program for a delivery problem, *Operations Research* **12**(2), 300–304.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P. and Vance, P. H. (1998), Branch-and-price: Column generation for solving huge integer programs, *Operations Research* **46**(3), 316–329.

Battarra, M., Cordeau, J.-F. and Iori, M. (2014), Chapter 6: Pickup-and-delivery problems for goods transportation, number 0 *in* 'MOS-SIAM Series on Optimization', Society for Industrial and Applied Mathematics, pp. 161–191.

Berbeglia, G., Cordeau, J.-F., Gribkovskaia, I. and Laporte, G. (2007), Static pickup and delivery problems: a classification scheme and survey, *TOP* **15**(1), 1–31.

Bianchessi, N. and Irnich, S. (2019), Branch-and-cut for the split delivery vehicle routing problem with time windows, *Transportation Science* **53**(2), 442–462.

Bruck, B. P., Cruz, F., Iori, M. and Subramanian, A. (2019), The static bike sharing rebalancing problem with forbidden temporary operations, *Transportation Science* **53**(3), 16.

Bulhões, T., Subramanian, A., Erdoğan, G. and Laporte, G. (2018), The static bike relocation problem with multiple vehicles and visits, *European Journal of Operational Research* **264**(2), 508 – 523.

Casazza, M., Ceselli, A., Chemla, D., Meunier, F. and Wolfler Calvo, R. (2019), The multiple vehicle balancing problem, *Networks* **72**(3), 337–357.

Casazza, M., Ceselli, A. and Wolfler Calvo, R. (2021), A route decomposition approach for the single commodity split pickup and split delivery vehicle routing problem, *European Journal of Operational Research* **289**(3), 897–911.

Chabrier, A. (2006), Vehicle routing problem with elementary shortest path based column generation, *Computers & Operations Research* **33**(10), 2972 – 2990. Part Special Issue: Constraint Programming.

Chemla, D., Meunier, F. and Wolfler Calvo, R. (2013), Bike sharing systems: Solving the static rebalancing problem, *Discrete Optimization* **10**(2), 120 – 146.

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

Cruz, F., Subramanian, A., Bruck, B. P. and Iori, M. (2017), A heuristic algorithm for a single vehicle static bike sharing rebalancing problem, *Computers & Operations Research* **79**, 19 – 33.

Dabia, S., Ropke, S., van Woensel, T. and De Kok, T. (2013), Branch and price for the time-dependent vehicle routing problem with time windows, *Transportation Science* **47**(3), 380–396.

Dell'Amico, M., Hadjicostantinou, E., Iori, M. and Novellani, S. (2014), The bike sharing rebalancing problem: Mathematical formulations and benchmark instances, *Omega* **45**, 7 – 19.

Dell'Amico, M., Iori, M., Novellani, S. and Stützle, T. (2016), A destroy and repair algorithm for the bike sharing rebalancing problem, *Computers & Operations Research* **71**, 149–162.

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., Desrosiers, J. and Solomon, M. M., eds (2005), *Column Generation*, Springer.

Dror, M. and Trudeau, P. (1989), Savings by split delivery routing, *Transportation Science* **23**(2), 141–145.

Erdoğan, G., Battarra, M. and Wolfler Calvo, R. (2015), An exact algorithm for the static rebalancing problem arising in bicycle sharing systems, *European Journal of Operational Research* **245**(3), 667 – 679.

Erdoğan, G., Laporte, G. and Wolfler Calvo, R. (2014), The static bicycle relocation problem with demand intervals, *European Journal of Operational Research* **238**(2), 451 – 457.

Espegren, H. M., Kristianslund, J., Andersson, H. and Fagerholt, K. (2016), The static bicycle repositioning problem - literature survey and new formulation, *in* A. Paias, M. Ruthmair and S. Voß, eds, 'Computational Logistics', Springer International Publishing, Cham, pp. 337 – 351.

Feillet, D., Dejax, P., Gendreau, M. and Gueguen, C. (2004), An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems, *Networks* **44**(3), 216 – 229.

Hernández-Pérez, H. and Salazar-González, J. (2004*a*), A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery, *Discrete Applied Mathematics* **145**(1), 126 – 139. Graph Optimization IV.

Hernández-Pérez, H. and Salazar-González, J. (2004*b*), Heuristics for the one-commodity pickup-and-delivery traveling salesman problem, *Transportation Science* **38**(2), 245 – 255.

Hernández-Pérez, H. and Salazar-González, J. (2007), The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms, *Networks* **50**(4), 258 – 272.

Hernández-Pérez, H., Salazar-González, J. and Santos-Hernández, B. (2018), Heuristic algorithm for the split-demand one-commodity pickup-and-delivery travelling salesman problem, *Computers & Operations Research* **97**, 1 – 17.

IBM CPLEX (2019), *IBM ILOG CPLEX 12.6.4 callable library.*

Irnich, S. and Desaulniers, G. (2005), Shortest path problems with resource constraints, *in* G. Desaulniers, J. Desrosiers and M. M. Solomon, eds, 'Column Generation', Springer, pp. 33 – 65.

Irnich, S., Schneider, M. and Vigo, D. (2014), Four variants of the vehicle routing problem, *in* P. Toth and D. Vigo, eds, 'Vehicle Routing', SIAM, chapter 9, pp. 241–271.

Irnich, S. and Villeneuve, D. (2006), The shortest-path problem with resource constraints andk-cycle elimination for $k \geq 3$, *INFORMS Journal on Computing* **18**(3), 391–406.

Izuno, T., Nishi, T. and Yin, S. (2012), Column generation for split pickup and delivery vehicle routing problem for crude oil transportation, *in* 'Proceedings of the ASME/ISCIE 2012 International Symposium on Flexible Automation', St. Louis, Missouri, USA, pp. 397 – 404.

Joncour, C., Michel, S., Sadykov, R., Sverdlov, D. and Vanderbeck, F. (2010), Column generation based primal heuristics, *ISCO 2010 - International Symposium on Combinatorial Optimization* **36**, 695–702.

Laporte, G., Meunier, F. and Wolfler Calvo, R. (2015), Shared mobility systems, *4OR* **13**(4), 341–360.

Laporte, G., Meunier, F. and Wolfler Calvo, R. (2018), Shared mobility systems: an updated survey, *Annals of Operations Research* **271**(1), 105–126.

Liberatore, F., Righini, G. and Salani, M. (2011), A column generation algorithm for the vehicle routing problem with soft time windows, *4OR* **9**(1), 49–82.

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

Luo, Z., Qin, H., Cheng, T. C. E., Wu, Q. and Lim, A. (2021), A branch-and-price-and-cut algorithm for the cable-routing problem in solar power plants, *INFORMS Journal on Computing* **33**(2), 419–835.

Luo, Z., Qin, H., Zhu, W. and Lim, A. (2017), Branch and price and cut for the split-delivery vehicle routing problem with time windows and linear weight-related cost, *Transportation Science* **51**(2), 668 – 687.

Parragh, S., Doerner, K. and Hartl, R. (2008*a*), A survey on pickup and delivery problems. Part I: Transportation between customers and depot, *Journal für Betriebswirtschaft* **58**(1), 21–51.

Parragh, S., Doerner, K. and Hartl, R. (2008*b*), A survey on pickup and delivery problems. Part II: Transportation between pickup and delivery locations, *Journal für Betriebswirtschaft* **58**(2), 81–117.

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

Poggi, M. and Uchoa, E. (2003), Integer program reformulation for robust branch-and-cut-and-price algorithms, *in* 'In Proceedings of the Conference Mathematical Program in Rio: A Conference in Honour of Nelson Maculan', pp. 56–61.

Poggi, M. and Uchoa, E. (2014), Chapter 3: New exact algorithms for the capacitated vehicle routing problem, number 0 *in* 'MOS-SIAM Series on Optimization', Society for Industrial and Applied Mathematics, pp. 59–86.

Righini, G. and Salani, M. (2006), Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints, *Discrete Optimization* **3**(3), 255 – 273.

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

Sadykov, R. and Vanderbeck, F. (2013), Bin packing with conflicts: A generic branch-and-price algorithm, *INFORMS Journal on Computing* **25**(2), 244–255.

Sadykov, R., Vanderbeck, F., Pessoa, A., Tahiri, I. and Uchoa, E. (2019), Primal heuristics for branch and price: The assets of diving methods, *INFORMS Journal on Computing* **31**(2), 251–267.

Salazar-González, J. and Santos-Hernández, B. (2015), The split-demand one-commodity pickup-and-delivery travelling salesman problem, *Transportation Research Part B: Methodological* **75**, 58 – 73.

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

Wei, L., Luo, Z., Baldacci, R. and Lim, A. (2020), A new branch-and-price-and-cut algorithm for one-dimensional bin-packing problems, *INFORMS Journal on Computing* **32**(2), 428–443.

This page is intentionally blank. Proper e-companion title page, with INFORMS branding and exact metadata of the main paper, will be produced by the INFORMS office when the issue is being assembled.

Proofs of statements, example, vehicle flow formulation and additional computational results

EC.1. Proofs of statements

This section gives the proofs of the different results of the main paper.

The proofs of Theorems 1 and 2 are based on the following lemma about the problem of minimizing a generic piecewise-linear convex function $f:\mathbb{R}_+\to\mathbb{R}$ expressed as $f(x)=\max_{h\in H}\{a_hx+b_h\}$, where $H=\{1,2,\ldots,m\}$ is the index set of its breakpoints, over an interval $[l,u]$, with $l\le u$, $l,u\in\mathbb{R}_+$. Let $H^-=\{h\in H\mid a_h<0\}$, and let $H^+=\{h\in H\mid a_h\ge 0\}$. Based on sets H^- and H^+, let functions $f^-:\mathbb{R}_+\to\mathbb{R}$ and $f^+:\mathbb{R}_+\to\mathbb{R}$ defined as $f^-(x)=\max_{h\in H^-}\{a_hx+b_h\}$, if $H^-\neq\emptyset$, $f^-(x)=-\infty$ otherwise, and $f^+(x)=\max_{h\in H^+}\{a_hx+b_h\}$, if $H^+\neq\emptyset$, $f^+(x)=-\infty$ otherwise. The following lemma holds.

LEMMA EC.1. *Let* $z^*=\min_x\{f(x)\}$. *Then* $\min_{l\le x\le u}\{f(x)\}=\max\{f^-(u),f^+(l),z^*\}$.

Proof. Let $x^*=\arg\min f(x)$. We have the following three cases.

(i) $u<x^*$. We have $f^-(u)\ge z^*\ge f^+(u)\ge f^+(l)$. Since $f(x)$ is a non-increasing function in $[l,u]$, we have $\min_{l\le x\le u}f(x)=f(u)=\max\{f^-(u),f^+(u)\}=f^-(u)=\max\{f^-(u),f^+(l),z^*\}$.

(ii) $l\le x^*\le u$. We have $z^*\ge f^-(u)$ and $z^*\ge f^+(l)$. Hence, $\min_{l\le x\le u}f(x)=z^*=\max\{f^-(u),f^+(l),z^*\}$.

(iii) $x^*<l$. We have $f^+(l)\ge z^*\ge f^-(l)\ge f^-(u)$. Since $f(x)$ is a non-decreasing function in $[l,u]$, we have $\min_{l\le x\le u}f(x)=f(l)=\max\{f^-(l),f^+(l)\}=f^+(l)=\max\{f^-(u),f^+(l),z^*\}$.

EC.1.1. Proof of Theorem 1

Given a forward path $P=(i_0=0,i_1,\ldots,i_{s-1},i_s)$, the proof is by the induction principle on s. Let $W(P)\le Q$ be the maximum amount of product collected from the pickup customers along path P that can be delivered to the delivery customers to be appended to the path. If $s=0$, i.e., $P=(0)$, we have $W(P)=0$ and $\overline{c}^f_P(w)=-\mu_0$, hence the hypothesis holds. Assume that for path $P=(i_0=0,i_1,\ldots,i_{s-1})$ the hypothesis holds, i.e., $\overline{c}^f_P(w)$ can be expressed in the form $\overline{c}^f_P(w)=\max_{h\in H}\{a_hw+b_h\}$, where H is the set of breakpoints and $a_h\ge 0$, $\forall h\in H$.

Consider a new vertex $i_s\notin V(P)$ appended at the end of path P and the corresponding path $P'=(i_0=0,i_1,\ldots,i_{s-1},i_s)$. We have two cases:

(i) Vertex i_s is a pickup vertex, i.e., $q_{i_s}>0$. We have $W(P')=\min\{Q,W(P)+q_{i_s}\}$, and

$$\begin{aligned}\overline{c}^f_{P'}(w)&=\min_{\max\{0,w-q_{i_s}\}\le x\le\min\{W(P),w\}}\{\overline{c}^f_P(x)-\mu_{i_s}(w-x)+d_{i_{s-1}i_s}\},\\&=\min_{\max\{0,w-q_{i_s}\}\le x\le\min\{W(P),w\}}\left\{\max_{h\in H}\{a_hx+b_h\}-\mu_{i_s}(w-x)+d_{i_{s-1}i_s}\right\},\\&=\min_{\max\{0,w-q_{i_s}\}\le x\le\min\{W(P),w\}}\left\{\max_{h\in H}\{(a_h+\mu_{i_s})x+b_h\}-\mu_{i_s}w+d_{i_{s-1}i_s}\right\}.\end{aligned}$$

Let $H^+ = \{h : h \in H, a_h + \mu_{i_s} \geq 0\}$, $H^- = \{h : h \in H, a_h + \mu_{i_s} < 0\}$, and $z^* = \min_x\{\max_{h\in H}\{(a_h+\mu_{i_s})x + b_h\}\}$. From Lemma EC.1, we have:

$$\begin{aligned}
\overline{c}^f_{P'}(w) &= \max\left\{\begin{array}{l} -\mu_{i_s}w + z^* + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\left\{(a_h+\mu_{i_s})\min\{W(P), w\} + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\}, \\ \max\limits_{h\in H^+}\left\{(a_h+\mu_{i_s})\max\{0, w-q_{i_s}\} + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\} \end{array}\right\}, \\
&= \max\left\{\begin{array}{l} -\mu_{i_s}w + z^* + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\left\{\max\{(a_h+\mu_{i_s})W(P), (a_h+\mu_{i_s})w\} + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\}, \\ \max\limits_{h\in H^+}\left\{(a_h+\mu_{i_s})\max\{0, w-q_{i_s}\} + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\} \end{array}\right\}, \\
&= \max\left\{\begin{array}{l} -\mu_{i_s}w + z^* + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\{(a_h+\mu_{i_s})W(P) + b_h\} - \mu_{i_s}w + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\{a_h w + b_h\} + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^+}\{b_h\} - \mu_{i_s}w + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^+}\{a_h w - (a_h+\mu_{i_s})q_{i_s} + b_h + d_{i_{s-1}i_s}\} \end{array}\right\}, \\
&= \max\left\{\begin{array}{l} -\mu_{i_s}w + \max\left\{z^*, \max\limits_{h\in H^-}\{(a_h+\mu_{i_s})W(P) + b_h\}, \max\limits_{h\in H^+}\{b_h\}\right\} + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\{a_h w + b_h\} + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^+}\{a_h w - (a_h+\mu_{i_s})q_{i_s} + b_h\} + d_{i_{s-1}i_s}, \end{array}\right\}.
\end{aligned}$$

Since $\mu_{i_s} \leq 0$, $\overline{c}^f_{P'}(w)$ is the maximum of piecewise-linear nondecreasing functions, hence it is a piecewise-linear convex nondecreasing function.

(ii) Vertex i_s is a delivery vertex, i.e., $q_{i_s} < 0$. We have $W(P') = W(P)$, and

$$\begin{aligned}
\overline{c}^f_{P'}(w) &= \min_{w\leq x\leq \min\{W(P), w-q_{i_s}\}}\{\overline{c}^f_P(x) - \mu_{i_s}(w-x) + d_{i_{s-1}i_s}\} \\
&= \min_{w\leq x\leq \min\{W(P), w-q_{i_s}\}}\left\{\max_{h\in H}\{a_h x + b_h\} - \mu_{i_s}(w-x) + d_{i_{s-1}i_s}\right\} \\
&= \min_{w\leq x\leq \min\{W(P), w-q_{i_s}\}}\left\{\max_{h\in H}\{(a_h+\mu_{i_s})x + b_h\} - \mu_{i_s}w + d_{i_{s-1}i_s}\right\}.
\end{aligned}$$

Let H^+, H^- and z^* be defined as for the previous case. We have:

$$\begin{aligned}
\overline{c}^f_{P'}(w) &= \max\left\{\begin{array}{l} -\mu_{i_s}w + z^* + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\left\{(a_h+\mu_{i_s})\min\{W(P), w-q_{i_s}\} + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\}, \\ \max\limits_{h\in H^+}\left\{(a_h+\mu_{i_s})w + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\} \end{array}\right\}, \\
&= \max\left\{\begin{array}{l} -\mu_{i_s}w + z^* + d_{i_{s-1}i_s}, \\ \max\limits_{h\in H^-}\left\{\max\{(a_h+\mu_{i_s})W(P), (a_h+\mu_{i_s})(w-q_{i_s})\} + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\}, \\ \max\limits_{h\in H^+}\left\{(a_h+\mu_{i_s})w + b_h - \mu_{i_s}w + d_{i_{s-1}i_s}\right\} \end{array}\right\},
\end{aligned}$$

$$= \max \left\{ \begin{array}{l} -\mu_{i_s} w + z^* + d_{i_{s-1} i_s}, \\ -\mu_{i_s} w + \max\limits_{h \in H^-} \{(a_h + \mu_{i_s}) W(P) + b_h\} + d_{i_{s-1} i_s}, \\ \max\limits_{h \in H^-} \{a_h w - (a_h + \mu_{i_s}) q_{i_s} + b_h + d_{i_{s-1} i_s}\}, \\ \max\limits_{h \in H^+} \left\{(a_h + \mu_{i_s}) w + b_h - \mu_{i_s} w + d_{i_{s-1} i_s}\right\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} -\mu_{i_s} w + \max\{z^*, \max\limits_{h \in H^-} \{(a_h + \mu_{i_s}) W(P) + b_h\}\} + d_{i_{s-1} i_s}, \\ \max\limits_{h \in H^-} \{a_h w - (a_h + \mu_{i_s}) q_{i_s} + b_h\} + d_{i_{s-1} i_s}, \\ \max\limits_{h \in H^+} \{a_h w + b_h\} + d_{i_{s-1} i_s} \end{array} \right\}.$$

Since $\mu_{i_s} \le 0$, $\overline{c}^f_{P'}(w)$ is also a piecewise-linear convex nondecreasing function.

EC.1.2. Proof of Theorem 2

Given a backward path $\overline{P} = (i_s, i_{s-1}, \ldots, i_1, i_0 = n')$, the proof is by the induction principle on s. Let $W(\overline{P}) \le Q$ be the maximum amount of the product required by the delivery customers along path $\overline{P}$ that can potentially be collected from the (pickup) customers, which is to be appended at the beginning of path $\overline{P}$.

If $s = 0$, i.e., $\overline{P} = (n')$, we have $W(\overline{P}) = 0$ and $\overline{c}^b_{\overline{P}}(w) = -\mu_0$, hence the hypothesis holds. Assume that for path $\overline{P} = (i_{s-1}, \ldots, i_1, i_0 = n')$ the hypothesis holds, that is, $\overline{c}^b_{\overline{P}}(w)$ can be expressed in the form $\overline{c}^b_{\overline{P}}(w) = \max_{h \in H}\{a_h w + b_h\}$, where H is the set of breakpoints and $a_h \le 0$, $\forall h \in H$.

Consider a new vertex $i_s \notin V(\overline{P})$ appended at the beginning of path $\overline{P}$ and the corresponding path $\overline{P}' = (i_s, i_{s-1}, \ldots, i_1, i_0 = n')$. We have two cases:

(i) Vertex i_s is a delivery vertex, that is, $q_{i_s} < 0$. We have $W(\overline{P}) = \min\{Q, W(\overline{P}) - q_{i_s}\}$, and

$$\begin{aligned} \overline{c}^b_{\overline{P}}(w) &= \min_{\max\{0, w+q_{i_s}\} \le x \le \min\{W(\overline{P}), w\}} \{\overline{c}^b_{\overline{P}}(x) - \mu_{i_s}(x - w) + d_{i_s i_{s-1}}\} \\ &= \min_{\max\{0, w+q_{i_s}\} \le x \le \min\{W(\overline{P}), w\}} \left\{ \max_{h \in H}\{a_h x + b_h\} - \mu_{i_s}(x - w) + d_{i_s i_{s-1}} \right\} \\ &= \min_{\max\{0, w+q_{i_s}\} \le x \le \min\{W(\overline{P}), w\}} \left\{ \max_{h \in H}\{(a_h - \mu_{i_s}) x + b_h\} + \mu_{i_s} w + d_{i_s i_{s-1}} \right\}. \end{aligned}$$

Let $H^+ = \{h : h \in H, a_h - \mu_{i_s} \ge 0\}$, $H^- = \{h : h \in H, a_h - \mu_{i_s} < 0\}$ and $z^* = \min_x \max_{h \in H}\{(a_h - \mu_{i_s}) x + b_h\}$. From Lemma EC.1, we have:

$$\overline{c}^b_{\overline{P}}(w) = \max \left\{ \begin{array}{l} \mu_{i_s} w + z^* + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \left\{(a_h - \mu_{i_s}) \min\{W(\overline{P}), w\} + b_h + \mu_{i_s} w + d_{i_s i_{s-1}}\right\}, \\ \max\limits_{h \in H^+} \left\{(a_h - \mu_{i_s}) \max\{0, w + q_{i_s}\} + b_h + \mu_{i_s} w + d_{i_s i_{s-1}}\right\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} \mu_{i_s} w + z^* + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \left\{\max\{(a_h - \mu_{i_s}) W(\overline{P}), (a_h - \mu_{i_s}) w\} + b_h + \mu_{i_s} w + d_{i_s i_{s-1}}\right\}, \\ \max\limits_{h \in H^+} \left\{(a_h - \mu_{i_s}) \max\{0, w + q_{i_s}\} + b_h + \mu_{i_s} w + d_{i_s i_{s-1}}\right\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} \mu_{i_s} w + z^* + d_{i_s i_{s-1}}, \\ \mu_{i_s} w + \max\limits_{h \in H^-} \{(a_h - \mu_{i_s}) W(\overline{P}) + b_h\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \{a_h w + b_h + d_{i_s i_{s-1}}\}, \\ \mu_{i_s} w + \max\limits_{h \in H^+} \{b_h\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^+} \{a_h w + (a_h - \mu_{i_s}) q_{i_s} + b_h + d_{i_s i_{s-1}}\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} \mu_{i_s} w + \max \left\{ z^*, \max\limits_{h \in H^-} \{(a_h - \mu_{i_s}) W(\overline{P}) + b_h\}, \max\limits_{h \in H^+} \{b_h\} \right\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \{a_h w + b_h\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^+} \{a_h w + (a_h - \mu_{i_s}) q_{i_s} + b_h\} + d_{i_s i_{s-1}} \end{array} \right\}.$$

Since $\mu_{i_s} \leq 0$, $\overline{c}^b_{\overline{P}'}(w)$ is the maximum of piecewise-linear nonincreasing functions, it is a piecewise-linear convex nonincreasing function.

(ii) Vertex i_s is a pickup vertex, i.e., $q_{i_s} > 0$. We have $W(\overline{P}') = W(\overline{P})$, and

$$\begin{aligned} \overline{c}^b_{\overline{P}}(w) &= \min_{w \leq x \leq \min\{W(\overline{P}), w + q_{i_s}\}} \{\overline{c}^b_{\overline{P}}(x) - \mu_{i_s}(x - w) + d_{i_s i_{s-1}}\} \\ &= \min_{w \leq x \leq \min\{W(\overline{P}), w + q_{i_s}\}} \left\{ \max_{h \in H} \{a_h x + b_h\} - \mu_{i_s}(x - w) + d_{i_s i_{s-1}} \right\} \\ &= \min_{w \leq x \leq \min\{W(\overline{P}), w + q_{i_s}\}} \left\{ \max_{h \in H} \{(a_h - \mu_{i_s}) x + b_h\} + \mu_{i_s} w + d_{i_s i_{s-1}} \right\}. \end{aligned}$$

Let H^+, H^- and z^* be defined as for the previous case. We have:

$$\overline{c}^b_{\overline{P}}(w) = \max \left\{ \begin{array}{l} \mu_{i_s} w + z^* + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \left\{ (a_h - \mu_{i_s}) \min\{W(\overline{P}), w + q_{i_s}\} + b_h + \mu_{i_s} w + d_{i_s i_{s-1}} \right\}, \\ \max\limits_{h \in H^+} \left\{ (a_h - \mu_{i_s}) w + b_h + \mu_{i_s} w + d_{i_s i_{s-1}} \right\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} \mu_{i_s} w + z^* + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \left\{ \max\{(a_h - \mu_{i_s}) W(\overline{P}), (a_h - \mu_{i_s})(w + q_{i_s})\} + b_h + \mu_{i_s} w + d_{i_s i_{s-1}} \right\}, \\ \max\limits_{h \in H^+} \left\{ (a_h - \mu_{i_s}) w + b_h + \mu_{i_s} w + d_{i_s i_{s-1}} \right\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} \mu_{i_s} w + z^* + d_{i_s i_{s-1}}, \\ \mu_{i_s} w + \max\limits_{h \in H^-} \{(a_h - \mu_{i_s}) W(\overline{P}) + b_h\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \{a_h w + (a_h - \mu_{i_s}) q_{i_s} + b_h + d_{i_s i_{s-1}}\}, \\ \max\limits_{h \in H^+} \left\{ a_h x + b_h + d_{i_s i_{s-1}} \right\} \end{array} \right\},$$

$$= \max \left\{ \begin{array}{l} \mu_{i_s} w + \max\{z^*, \max\limits_{h \in H^-} \{(a_h - \mu_{i_s}) W(\overline{P}) + b_h\}\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^-} \{a_h w + (a_h - \mu_{i_s}) q_{i_s} + b_h\} + d_{i_s i_{s-1}}, \\ \max\limits_{h \in H^+} \{a_h w + b_h\} + d_{i_s i_{s-1}} \end{array} \right\}.$$

Since $\mu_{i_s} \leq 0$, $\overline{c}^b_{P'}(w)$ is also a piecewise-linear convex nonincreasing function.

EC.1.3. Proof of Dominance 1

Let P be the forward path associated with label L^f_1, and we denote by $\{L^f_{i_1}, L^f_{i_2}, \ldots, L^f_{i_k}\}$ the set of labels $\mathscr{L}^f_1$, where a forward path P_{i_s} is associated with label $L^f_{i_s}$, $s = 1, \ldots, k$.

Consider the set $\mathcal{P}$ of all feasible extensions of path P, i.e., the set of backward paths starting at vertices different from vertex v^f_1 such that for each $\overline{P} \in \mathcal{P}$ route $R = (P, \overline{P})$ is feasible for the maximum duration constraint. Let $w(P)$, $w(P) \in [0, \min\{W^f_1, W(\overline{P})\}]$, be the load of the vehicle leaving the last vertex v^f_1 visited by route R where $W(\overline{P}) \leq Q$ is the maximum amount of the product required by the delivery customers along path $\overline{P}$, and let j be the first vertex visited by path $\overline{P}$. The reduced cost of route R and value $w(P)$ can be computed as follows:

$$\overline{c}(R, w(P)) = \overline{c}^f_P(w(P)) + d_{v^f_1 j} + \overline{c}^b_{\overline{P}}(w(P)).$$

Below, we show that for any path $\overline{P} \in \mathcal{P}$ and associated value $w(P)$, there exists a forward path $P' \in \{P_{i_s}\}^k_{s=1}$ such that route $R' = (P', \overline{P})$ is feasible and its reduced cost $\overline{c}(R', w(P))$ is less than or equal to the reduced cost $\overline{c}(R, w(P))$ of route R, and thus path P can be safely discarded.

Given a path $\overline{P} \in \mathcal{P}$, since $v^f = v^f_1$, $s^f \geq s^f_1$, and $V^f_1 \subseteq V^f$, for all $L^f \in \mathscr{L}^f_1$, each route $R' = (P', \overline{P})$ with $P' \in \{P_{i_s}\}^k_{s=1}$ is feasible for the maximum duration constraint. Let $i^* = \arg\min_{s=1,\ldots,k}\{\overline{c}^f_{P_{i_s}}(w(P)\}$ be the index of the labels in L^f_1 having minimum reduced cost computed with respect to value $w(P)$.

Since $w(P) \leq W^f_1$, we have $g_{L^f_1}(w(P)) = \overline{c}^f_P(w(P)) < +\infty$, and given the definition of function $\overline{g}_{\mathscr{L}^f_1}(w)$ we have $\overline{g}_{\mathscr{L}^f_1}(w(P)) = \overline{c}^f_{P_{i^*}}(w(P))$. Consider route $R' = (P_{i^*}, \overline{P})$. The reduced cost of route R' can be computed as

$$\overline{c}(R', w(P)) = \overline{c}^f_{P_{i^*}}(w(P)) + d_{v^f_1 j} + \overline{c}^b_{\overline{P}}(w(P)) = \overline{g}_{\mathscr{L}^f_1}(w(P)) + d_{v^f_1 j} + \overline{c}^b_{\overline{P}}(w(P)). \tag{EC.1}$$

From the hypothesis we have $g_{L^f_1}(w(P)) \geq \overline{g}_{\mathscr{L}^f_1}(w(P))$, and thus $\overline{g}_{\mathscr{L}^f_1}(w(P)) < +\infty$ and $w(P) \in [0, W^f_{i^*}]$, so from expression (EC.1) we obtain:

$$\begin{aligned}
\overline{c}(R', w(P)) = \overline{g}_{\mathscr{L}^f_1}(w(P)) + d_{v^f_1 j} + \overline{c}^b_{\overline{P}}(w(P)) \leq \\
g_{L^f_1}(w(P)) + d_{v^f_1 j} + \overline{c}^b_{\overline{P}}(w(P)) = \\
\overline{c}^f_P(w(P)) + d_{v^f_1 j} + \overline{c}^b_{\overline{P}}(w(P)) = \\
\overline{c}(R, w(P)),
\end{aligned}$$

and thus $\overline{c}(R', w(P)) \leq \overline{c}(R, w(P))$ and path P and the associated label L^f_1 can be safely discarded.

Table EC.1 Example of a fractional θ_{pr} solution satisfying the four branching rules

Variables θ_{pr}	Routes and demand patterns (in brackets)
0.5	0(0) 10(1) 3(-1) 8(3) 1(-3) 11(0)
1.0	0(0) 6(1) 9(-1) 4(6) 9(-4) 5(-2) 2(5) 5(-5) 1(0) 11(0)
0.5	0(0) 10(1) 3(-1) 8(6) 7(-6) 8(6.0) 7(-6.0) 8(3) 1(-3) 11(0)

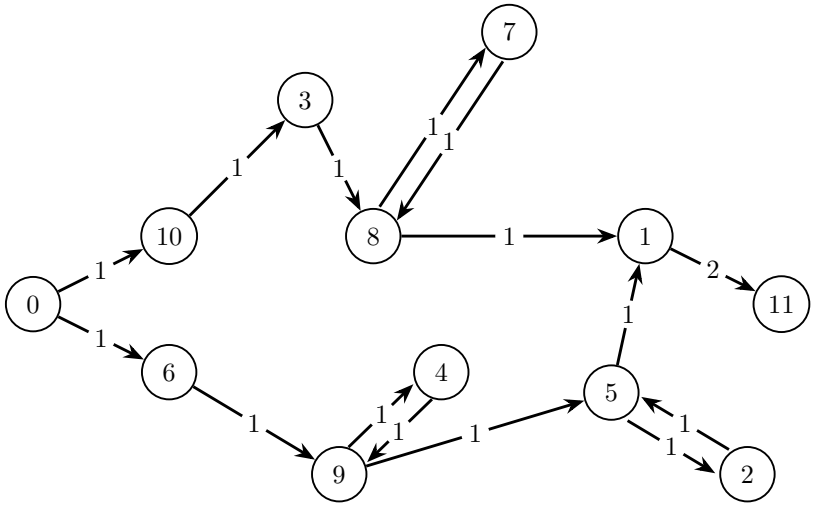

Figure EC.1 Support graph for the example of Table EC.1

EC.2. Example of a fractional RMP solution satisfying the four branching rules

Figure EC.1 shows an example of a fractional RMP solution satisfying the four branching rules. The example involves an $n = 10$ customers instance (numbered from 1 to 10) with initial and final depots indexed with numbers 0 and 11, respectively. The customers demands are $d_1 = -3$, $d_2 = 5$, $d_3 = -1$, $d_4 = 6$, $d_5 = -7$, $d_6 = 1$, $d_7 = -6$, $d_8 = 9$, $d_9 = -5$, $d_{10} = 1$, and the vehicle capacity is $Q = 6$. The routes and demand patterns of the solution are shown in Table EC.1, where the numbers in parentheses show the demand patterns. The support graph of the solution associated with values $\overline{\omega}_{ij}$ is depicted in Figure EC.1, where the number on each arc indicates the value of the corresponding variable $\overline{\omega}_{ij}$.

EC.3. A three-index (TI) vehicle flow formulation

The three-index (TI) vehicle flow formulation is based on three-index vehicle flow formulations proposed for the basic Capacitated VRP (CVRP) (see, for example, Toth and Vigo 2014) and on the single commodity flow formulation described in Salazar-González and Santos-Hernández (2015). In this section, we describe the formulation for the multiple-visit, different-vehicle case, and then we briefly observe how the formulation can be extended to deal with the multiple-visit case.

For a given $S \subseteq N$ we denote by $A(S)$ the set of arcs with both end-vertices in S, i.e., $A(S) = \{(i,j) \in A : i,j \in S\}$. Let x_{ij}^k be a binary variable that is equal to 1 if vehicle k traverses arc (i,j),

and 0 otherwise, and y_i^k be a continuous variable representing the amount of products picked up (if positive) or delivered (if negative) to vertex i by vehicle k. In addition, let z_i^k be a nonnegative variable representing the amount of product carried by vehicle k when it leaves vertex i. The formulation is as follows.

$$(TI) \quad \min \sum_{k\in K}\sum_{(i,j)\in A} d_{ij}x_{ij}^k \tag{EC.2a}$$

$$\text{s.t.} \sum_{(0,i)\in A} x_{0i}^k = \sum_{(i,n')\in A} x_{in'}^k, \quad \forall\ k\in K, \tag{EC.2b}$$

$$\sum_{(i,j)\in A} x_{ij}^k = \sum_{(j,i)\in A} x_{ji}^k, \quad \forall\ i\in V,\ k\in K, \tag{EC.2c}$$

$$\sum_{k\in K} y_i^k = q_i, \quad \forall\ i\in V, \tag{EC.2d}$$

$$y_i^k \le \min\{q_i, Q\} \sum_{(i,j)\in A} x_{ij}^k, \quad \forall\ i\in N^+,\ k\in K, \tag{EC.2e}$$

$$y_i^k \ge \max\{q_i, -Q\} \sum_{(i,j)\in A} x_{ij}^k, \quad \forall\ i\in N^-,\ k\in K, \tag{EC.2f}$$

$$z_j^k \ge z_i^k + y_j^k + (x_{ij}^k - 1)(Q+1), \quad \forall\ j\in N^+, (i,j)\in A,\ k\in K, \tag{EC.2g}$$

$$z_j^k \le z_i^k + y_j^k + (1 - x_{ij}^k)(Q+1), \quad \forall\ j\in N^-, (i,j)\in A,\ k\in K, \tag{EC.2h}$$

$$z_i^k \le Q, \quad \forall\ i\in V,\ k\in K, \tag{EC.2i}$$

$$\sum_{(i,j)\in A} t_{ij}x_{ij}^k \le T, \quad \forall\ k\in K, \tag{EC.2j}$$

$$\sum_{(i,j)\in A(S)} x_{ij}^k \le |S| - 1, \quad \forall\ S\subseteq N, |S|>1,\ k\in K, \tag{EC.2k}$$

$$y_0^k = y_{n'}^k = 0, \quad \forall\ k\in K, \tag{EC.2l}$$

$$x_{ij}^k \in \{0,1\}, \quad \forall\ (i,j)\in A,\ k\in K, \tag{EC.2m}$$

$$y_i^k \ge 0, \quad \forall\ i\in N^+,\ k\in K, \tag{EC.2n}$$

$$y_i^k \le 0, \quad \forall\ i\in N^-,\ k\in K, \tag{EC.2o}$$

$$z_i^k \ge 0, \quad \forall\ i\in V, k\in K. \tag{EC.2p}$$

The objective function (EC.2a) aims at minimizing the total routing cost of the vehicles. Constraints (EC.2b) and (EC.2c) are the flow conservation constraints for each vehicle at the depot and the customers, respectively. The satisfaction of the demand of each customer is guaranteed by constraints (EC.2d). Constraints (EC.2e) and (EC.2f) state that a vehicle can pick up or deliver the product at a customer only if the customer is visited by the vehicle, respectively. Constraints (EC.2g) and (EC.2h) define the values of flow variables z for the pickup and delivery customers, respectively. Constraints (EC.2i) and (EC.2j) are the capacity and the maximum duration constraints, respectively. Constraints (EC.2k) are the *subtour elimination* constraints.

Equations (EC.2l) impose that each vehicle departs from and arrives at the depot empty. Finally, constraints (EC.2m), (EC.2n), (EC.2o) and (EC.2p) state the domains of the decision variables.

Due to the presence of constraints (EC.2k), formulation TI imposes that a customer can be visited by the same vehicle at most once. Nevertheless, the formulation can be extended to deal with the case where a customer can also be visited more than once by the same vehicle, thus modeling the multiple visits case. Indeed, as done for example by Salazar-González and Santos-Hernández (2015) and Bulhões et al. (2018), multiple visits by the same vehicle can be modeled on an extended network where each vertex is associated with a number of vertices (representing possible visits) and each vertex can be visited at most once. A drawback of the resulting mathematical formulation is the large number of variables required by the underlying extended formulation, and for this reason the maximum number of visits to each vertex is generally fixed to a small value, such as less than two or three (see, for example, Salazar-González and Santos-Hernández 2015, Bulhões et al. 2018). The corresponding details are omitted for sake of brevity.

EC.4. Additional computational details

EC.4.1. Branching rules

Figures EC.2 and EC.3 give an overview of the use of the different branching rules adopted in EXM (see §5.3) for S_A, S_B and S_C classes, respectively. The figures show the percentages of the nodes in which each rule is used over the total number of branching decisions. In Figure EC.2, the instances are grouped by the vehicle capacity Q and the maximum duration T, and the corresponding pairs are reported in the figures as $[Q,T]$. The corresponding detailed values can be found in the tables given in the e-companion.

The figures clearly show that branching on the number of vehicles visiting each customer (Rule 2) and branching on single arc (Rule 3) are among the most used rules. Rule 4, that is, branching on two consecutive arcs, is rarely used. Indeed, it is applied 16 and 14 times for classes S_A, S_B and S_C, respectively (the corresponding percentages cannot be seen from the figures). As discussed in Section 5.3, the four branching rules are not sufficient to fulfil the integrality requirements. Nevertheless, in our experiments, it never once occurred that none of the rules could be applied.

EC.4.2. Analysis of the SPDVRP solutions

In this section, we report some insights into the solutions computed by EXM regarding the number of routes and the type of routes generated.

Figures EC.4 and EC.5 reports the average number of routes in the optimal or best solutions found of classes S_A, S_B and S_C, respectively. The data are grouped by the values of Q and T and the number of customers. The figures show that the average number of routes ranges between 1 and about 8. The numbers for different values of T show that the maximum duration constraints

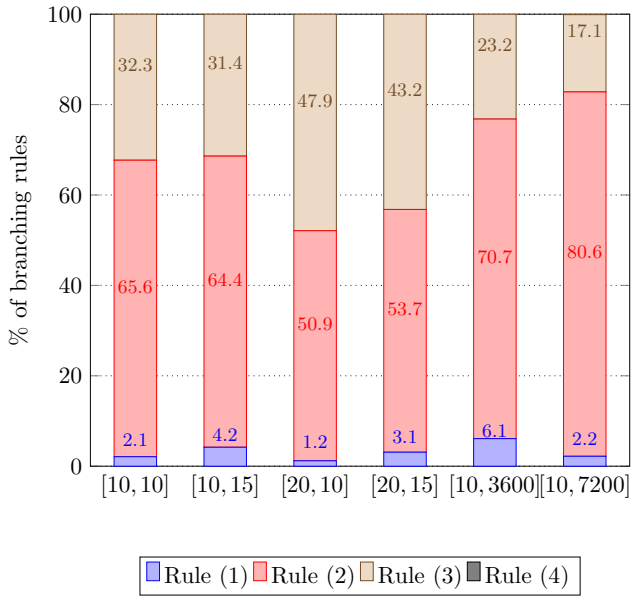

Figure EC.2 **Usage of branching decisions on S_A and S_B classes (values $[Q, T]$ under each bar)**

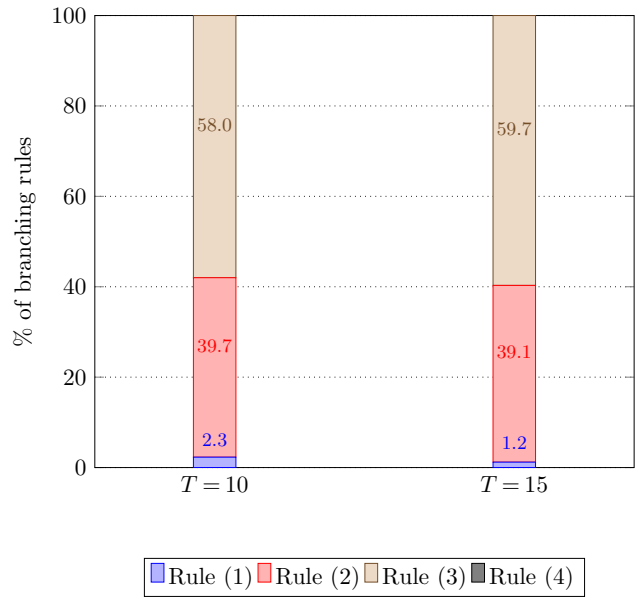

Figure EC.3 **Usage of branching decisions on S_C class**

(either as a cardinality or time constraint) are generally tight, as shown by the lower average numbers for increasing values of T.

Finally, Figures EC.6 and EC.7 give an overview of the structure of the solutions computed by EXM based on the way the customers are visited. More specifically, we denote by "sv" the total number of customers visited more than once by the same vehicle, and we denote by "dv" the total number of customers visited more than once by different vehicles. Parameters sv and dv can also be used as a measure of the difficulty of the instances. Figure EC.6 shows relevant data about

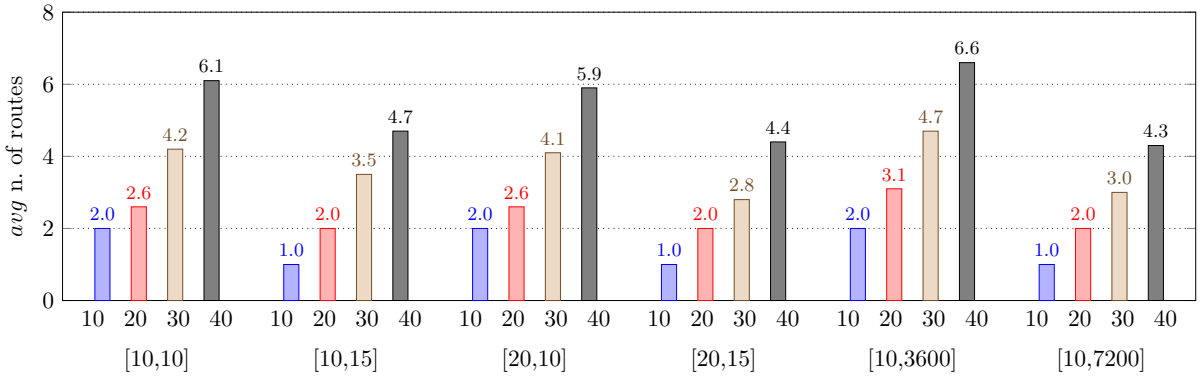

Figure EC.4 **Number of routes for the instances of classes S_A and S_B grouped by values $[Q,T]$ and number of customers**

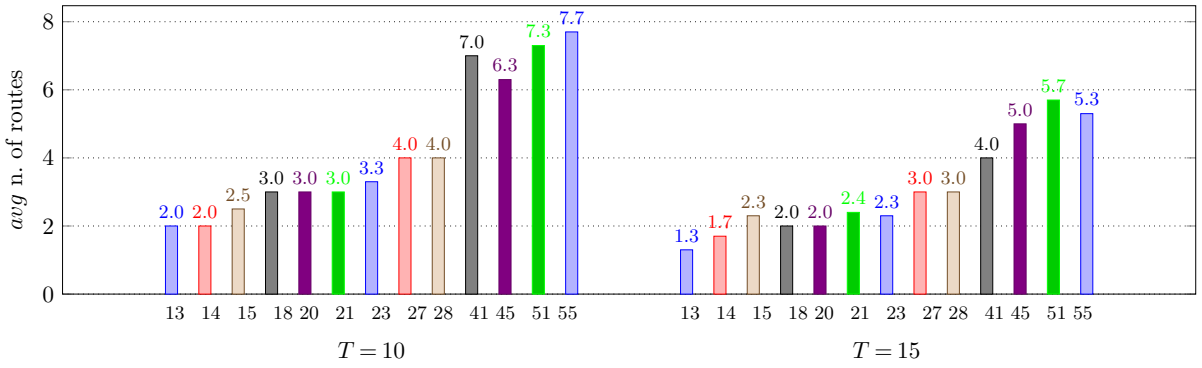

Figure EC.5 **Number of routes for the instances of class S_C grouped by the values of T and number of customers**

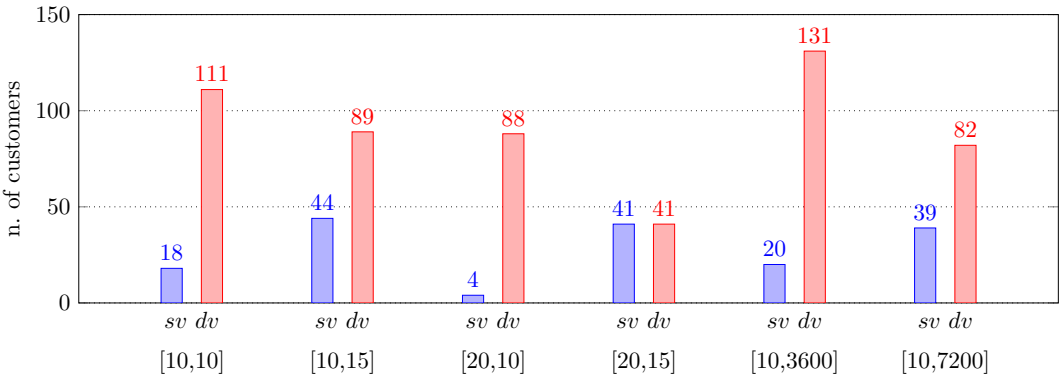

Figure EC.6 **Classes S_A and S_B: structure of the solutions grouped by values $[Q,T]$**

classes S_A and S_B, where the instances are grouped by values Q and T, whereas Figure EC.7 gives the details about class S_C, grouped by $T = 10$ and $T = 15$.

The figures clearly point out that the solutions computed (which include both optimal and heuristic solutions) feature a higher number of customers visited by different vehicles (measure dv) than the number of customers visited by the same vehicle (measure sv), and that instances with tight Q and T values generally involve more split customers. The comparison between classes S_A,

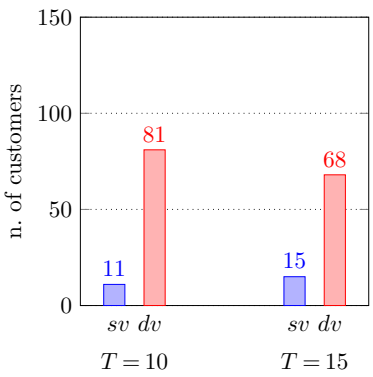

Figure EC.7 **Class S_C: structure of the solutions grouped by values of T**

S_B and S_C shows that class S_C features a reduced number of split customers with respect to classes S_A and S_B, that is to say that the routes forming the solutions of class S_C are generally elementary and disjoint. In this case, the literature shows that for classical (non-split) vehicle routing problems, the LP-relaxation of the set-partitioning formulation provides very tight lower bounds (see Poggi and Uchoa 2014). These differences could explain the fact that, according to Table 3, EXM shows tighter bounds for class S_C.

EC.5. Detailed computational results

This section reports detailed results about algorithm EXM on classes of instances S_A, S_B and S_C.

Tables EC.2-EC.9 show the following details:

- Name of the instance ("Name").
- Number of vehicles available ("$|K|$").
- Number of customers ("n").
- Vehicle capacity ("Q").
- Cost of the best solution found ("z^*") by EXM.
- Number of routes composing the solution ("rt").
- Percentage deviation of the upper bound computed by the primal heuristic ("$\%UB$"), computed as $100.0 \times UB/z^*$ where UB is the value of the upper bound.
- Time in seconds spent by the primal heuristic ("t_{UB}").
- Percentage deviation of lower bound derived from the LP-relaxation of formulation $\overline{PB}$ ("$\%LB$"), computed as $100.0 \times LB/z^*$ where LB is the value of the lower bound.
- Percentage deviation of lower bound obtained from the column-and-row generation procedure ("$\%LB_r$"), computed as $100.0 \times LB_r/z^*$ where LB_r is the value of the lower bound.
- Number of cuts generated at the root node ("cut_r").
- Time in seconds spent by the pricing algorithm at the root node ("t_{PA}").
- Time in seconds spent to compute the lower bound at the root node ("t_r").

- Total number of cuts generated ("*cut*").
- Number of tree nodes explored ("*node*").
- Total computing time in seconds ("t").

EC.6. Detailed computational results about the pricing algorithms

This section reports detailed results about the comparison between the pricing algorithm of EXM and our implementation of the pricing algorithm proposed by Casazza et al. (2019).

Tables EC.10-EC.12 show the following details:

- Name of the instance ("Name").
- Number of customers ("n").
- Number of vehicles available ("$|K|$").
- Lower bound value ("LB").
- For each pricing algorithm:
  - Time in seconds spent by the LP solver ("t_{LP}").
  - Time in seconds spent by the pricing algorithm ("t_{PA}").
  - Number of forward labels generated ("fw").
  - Number of forward labels dominated ("fwd").
  - Number of backward labels generated ("bw").
  - Number of backward labels dominated ("bwd").

Table EC.2 Detailed results on set S_A with $Q = 10$ and $T = 10$

Name	n	$\|K\|$	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
n10a	10	5	3719.0	2	105.2	0.8	86.3	95.2	63	0.2	0.4	67	27	1.5
n10A	10	5	3055.0	2	100.0	0.6	95.6	99.8	56	0.2	0.4	56	3	1.0
n10b	10	5	3192.0	2	100.0	0.8	84.2	99.8	24	0.3	0.4	24	3	1.3
n10B	10	5	3704.0	2	101.5	0.8	95.6	99.6	61	0.3	0.5	61	3	1.6
n10c	10	5	4239.0	2	100.0	0.4	98.7	100.0	8	0.2	0.3	8	1	0.9
n10C	10	5	3392.0	2	105.3	0.9	84.9	97.7	101	0.4	0.6	102	11	1.8
n10d	10	5	4497.0	2	102.4	0.7	86.7	96.5	28	0.3	0.4	34	35	2.0
n10D	10	5	3199.0	2	111.2	0.8	81.6	97.3	60	0.3	0.5	60	3	1.3
n10e	10	5	3823.0	2	100.0	0.5	89.0	100.0	87	0.4	0.6	87	1	1.2
n10E	10	5	4876.0	2	100.0	0.5	92.1	100.0	85	0.3	0.5	85	1	1.1
n10f	10	5	3468.0	2	126.4	1.0	76.6	86.0	65	0.4	0.6	109	1635	81.7
n10F	10	5	3796.0	2	100.0	0.6	96.5	99.8	53	0.2	0.4	53	3	1.3
n10g	10	5	4179.0	2	107.7	0.9	81.8	85.8	58	0.2	0.4	75	1953	102.8
n10G	10	5	3973.0	2	139.5	0.6	86.5	90.1	28	0.2	0.4	28	3	1.1
n10h	10	5	4168.0	2	100.0	0.6	88.4	96.3	66	0.2	0.4	67	13	1.3
n10H	10	5	3959.0	2	108.0	0.9	80.7	93.9	65	0.3	0.5	73	155	7.3
n10i	10	5	2645.0	2	145.0	0.7	91.7	95.6	56	0.2	0.4	58	9	1.3
n10I	10	5	3963.0	2	133.9	0.6	83.9	86.8	25	0.2	0.4	31	35	1.6
n10j	10	5	3453.0	2	115.0	0.6	94.0	94.6	19	0.2	0.4	21	17	1.2
n10J	10	5	3125.0	2	134.3	0.7	79.4	98.6	56	0.2	0.4	57	17	1.2
n20A	20	5	4826.0	2	115.7	2.5	94.0	99.3	172	0.6	1.0	176	13	4.2
n20B	20	5	5300.0	2	108.3	2.7	85.5	94.6	155	0.9	1.3	173	135	21.9
n20C	20	5	6508.0	3	115.3	4.9	92.0	96.2	486	1.3	1.9	604	8775	4591.5
n20D	20	5	6208.0	3	100.0	2.9	96.7	100.0	171	0.7	1.0	171	3	4.2
n20E	20	5	6491.0	3	114.2	4.1	88.6	98.2	367	1.1	1.7	401	261	85.7
n20F	20	5	5222.0	3	120.8	3.2	92.7	95.5	161	0.9	1.3	173	33	9.1
n20G	20	5	5795.0	3	122.6	3.3	86.0	94.3	190	1.0	1.3	277	1147	161.7
n20H	20	5	6207.0	2	127.8	4.1	85.8	93.1	340	1.0	1.5	358	547	77.8
n20I	20	5	5167.0	3	113.7	3.3	85.9	94.1	210	0.9	1.3	281	3521	789.5
n20J	20	5	4545.0	2	108.1	1.6	92.1	96.3	120	0.5	0.8	161	155	17.5
n30A	30	5	6865.0	4	105.0	15.0	92.5	97.3	1038	6.7	5.7	1252	5305	4637.3
n30B	30	5	6939.0	3	108.5	5.5	91.2	97.7	387	1.9	2.6	454	445	237.1
n30C	30	5	8613.0	5	100.0	16.6	73.0	79.3	594	8.5	3.2	1221	16991	10800.0
n30D	30	5	9314.0	5	100.0	8.2	66.6	71.4	534	2.3	2.8	923	16603	10800.0
n30E	30	5	6840.0	3	114.8	7.8	86.3	94.7	486	1.9	2.4	778	1039	1095.2
n30F	30	5	6715.0	4	106.2	7.5	88.4	95.4	319	2.2	2.9	502	2735	943.8
n30G	30	5	9462.0	4	105.3	12.8	90.7	97.0	778	3.3	4.5	1228	13401	10360.7
n30H	30	5	8182.0	5	100.0	7.3	75.1	79.8	452	2.4	3.0	943	17257	10800.0
n30I	30	5	6602.0	4	100.0	5.8	82.8	87.9	335	1.6	2.2	788	18217	10800.0
n30J	30	5	8753.0	5	100.0	9.0	70.7	73.2	365	3.2	2.1	1702	19701	10800.0
n40A	40	8	10430.0	6	100.0	18.7	71.5	74.4	789	3.4	4.2	1604	14157	10800.0
n40B	40	8	9805.0	7	100.0	18.9	69.1	72.4	736	3.1	3.7	1334	14171	10800.0
n40C	40	8	9097.0	5	100.0	12.4	81.4	85.9	818	3.3	4.4	1201	5011	10800.0
n40D	40	8	9837.0	5	100.0	19.7	80.9	86.2	964	3.9	5.0	1387	11993	10800.0
n40E	40	8	11882.0	7	100.0	13.9	55.8	61.2	792	2.4	3.0	1409	11447	10800.0
n40F	40	8	9275.0	5	100.0	15.8	79.5	85.2	711	3.4	4.2	1265	12583	10800.0
n40G	40	8	11017.0	7	100.0	16.5	73.7	76.7	878	3.6	4.5	1593	13383	10800.0
n40H	40	8	10129.0	7	100.0	15.3	72.9	75.0	869	3.0	3.6	1183	13475	10800.0
n40I	40	8	11222.0	6	100.0	14.4	66.1	70.2	798	3.1	3.9	1285	13119	10800.0
n40J	40	8	9265.0	6	100.0	11.8	73.8	77.4	574	2.5	3.1	1538	14155	10800.0

Table EC.3 Detailed results on set S_A with $Q = 10$ and $T = 15$

Name	n	$\|K\|$	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
n10a	10	5	3484.0	1	148.6	1.2	90.3	100.0	52	0.5	0.7	52	3	2.0
n10A	10	5	2994.0	1	100.0	0.2	96.4	100.0	28	0.1	0.1	28	1	0.5
n10b	10	5	3122.0	1	100.0	0.4	84.7	100.0	24	0.1	0.2	24	1	0.6
n10B	10	5	3671.0	1	100.0	0.4	96.1	100.0	60	0.1	0.2	60	1	0.6
n10c	10	5	4226.0	1	100.0	0.1	98.0	100.0	16	0.0	0.0	16	1	0.2
n10C	10	5	3324.0	1	135.4	1.0	84.4	99.2	136	0.3	0.4	138	9	1.5
n10d	10	5	4238.0	1	100.0	0.3	89.0	100.0	28	0.1	0.1	28	1	0.5
n10D	10	5	3071.0	1	100.0	0.5	80.0	100.0	96	0.3	0.4	96	1	0.5
n10e	10	5	3816.0	1	100.0	0.5	88.3	100.0	77	0.2	0.2	77	1	0.2
n10E	10	5	4828.0	1	100.0	0.5	90.6	100.0	84	0.2	0.3	84	1	0.2
n10f	10	5	2996.0	1	170.4	0.6	83.4	98.0	67	0.2	0.3	68	13	2.7
n10F	10	5	3758.0	1	100.0	0.3	94.2	100.0	68	0.1	0.2	68	1	0.5
n10g	10	5	3685.0	1	134.9	0.3	85.8	93.1	61	0.1	0.1	71	359	43.9
n10G	10	5	3325.0	1	100.0	0.2	92.8	100.0	16	0.1	0.1	16	1	0.4
n10h	10	5	3960.0	1	126.3	0.3	87.8	99.8	58	0.1	0.1	58	3	0.7
n10H	10	5	3813.0	1	183.5	1.0	79.9	95.3	69	0.3	0.4	92	695	111.7
n10i	10	5	2390.0	1	100.0	0.2	96.4	100.0	23	0.1	0.1	23	1	0.4
n10I	10	5	3287.0	1	174.7	0.3	92.4	98.2	35	0.1	0.2	35	5	1.5
n10j	10	5	3249.0	1	122.3	0.2	97.5	98.9	44	0.1	0.1	44	11	1.6
n10J	10	5	3060.0	1	100.0	0.2	70.8	100.0	40	0.1	0.1	40	1	0.4
n20A	20	5	4713.0	2	148.2	2.9	90.7	99.2	178	1.0	1.3	179	9	8.9
n20B	20	5	4992.0	2	102.6	3.9	83.9	96.8	191	1.6	1.9	203	91	45.1
n20C	20	5	6110.0	2	133.4	12.9	93.6	99.0	680	5.0	5.8	702	371	2193.8
n20D	20	5	6146.0	2	103.4	6.8	93.0	99.3	372	2.0	2.5	396	135	155.4
n20E	20	5	6339.0	2	115.3	12.9	88.3	99.0	562	5.4	6.4	589	101	171.2
n20F	20	5	4851.0	2	157.2	3.3	93.0	97.3	258	1.1	1.3	272	25	19.9
n20G	20	5	5334.0	2	101.8	3.9	87.1	97.3	201	1.4	1.7	203	9	14.1
n20H	20	5	5847.0	2	140.1	5.2	84.7	95.1	327	1.7	2.1	405	1967	863.9
n20I	20	5	4797.0	2	125.4	7.3	85.8	97.3	483	1.5	2.1	529	307	263.7
n20J	20	5	4402.0	2	117.0	1.6	90.2	95.2	184	0.3	0.4	193	39	8.7
n30A	30	5	8367.0	4	100.0	54.2	70.1	75.5	1631	17.3	20.8	1796	2907	10800.0
n30B	30	5	6460.0	2	109.3	15.0	92.6	99.4	826	5.7	7.2	831	65	436.8
n30C	30	5	7676.0	4	100.0	42.7	74.7	83.0	1099	9.1	10.7	1361	6651	10800.0
n30D	30	5	7403.0	4	100.0	48.9	73.7	82.6	1242	10.5	11.3	1757	6345	10800.0
n30E	30	5	6286.0	2	126.3	19.2	83.9	94.9	632	4.5	5.3	1141	4065	10800.0
n30F	30	5	6205.0	3	124.6	19.3	86.4	94.4	556	7.4	9.1	1012	8487	10800.0
n30G	30	5	9163.0	3	117.6	64.1	88.8	97.1	1270	15.8	18.8	1738	3347	10800.0
n30H	30	5	9191.0	5	100.0	16.0	60.7	65.0	617	4.5	5.4	1249	6897	10800.0
n30I	30	8	7510.0	5	100.0	12.4	65.2	71.6	713	3.5	4.9	868	4565	10800.0
n30J	30	8	6204.0	3	113.6	11.4	89.7	95.3	638	2.7	3.4	1000	3351	4142.6
n40A	40	8	12046.0	6	100.0	44.5	56.2	59.2	1204	10.5	12.7	1566	3811	10800.0
n40B	40	8	7989.0	4	100.0	37.9	73.2	78.9	780	6.3	7.1	1193	5771	10800.0
n40C	40	8	9431.0	5	100.0	71.8	72.1	78.1	1395	26.9	31.5	1525	1895	10800.0
n40D	40	8	8354.0	3	100.0	99.2	87.1	95.7	991	35.0	38.3	1187	1511	10800.0
n40E	40	8	11064.0	6	100.0	34.1	51.9	59.4	805	7.4	8.8	1166	4267	10800.0
n40F	40	8	9281.0	5	100.0	71.1	71.4	78.3	1395	17.5	19.7	1788	2557	10800.0
n40G	40	8	8150.0	4	104.4	31.1	88.8	93.9	800	9.5	11.1	1263	2953	10800.0
n40H	40	8	9224.0	5	100.0	34.5	70.3	73.2	945	9.4	11.5	1157	4329	10800.0
n40I	40	8	10135.0	4	100.0	45.5	64.8	71.2	879	10.7	11.9	1561	3809	10800.0
n40J	40	8	10608.0	5	100.0	35.7	56.1	61.5	1011	5.0	5.9	1313	5747	10800.0

Table EC.4 Detailed results on set S_A with $Q = 20$ and $T = 10$

Name	n	$\|K\|$	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
n10a	10	5	3524.0	2	111.6	0.7	86.4	94.5	31	0.3	0.4	33	13	1.4
n10A	10	5	2936.0	2	117.6	0.6	84.7	87.3	24	0.3	0.4	24	5	1.1
n10b	10	5	3192.0	2	115.8	0.7	82.4	92.6	13	0.3	0.4	13	3	2.8
n10B	10	5	3587.0	2	100.8	0.8	86.4	88.5	36	0.3	0.4	36	3	1.2
n10c	10	5	3324.0	2	115.3	0.7	94.9	94.9	0	0.3	0.5	0	3	1.9
n10C	10	5	3013.0	2	118.2	1.0	82.0	91.1	32	0.3	0.5	32	3	2.4
n10d	10	5	4043.0	2	109.5	0.7	86.3	88.5	24	0.3	0.5	24	43	3.8
n10D	10	5	2988.0	2	104.0	0.7	84.6	97.9	23	0.3	0.4	23	3	1.3
n10e	10	5	3130.0	2	117.7	1.5	87.8	92.4	38	0.4	0.7	39	25	3.0
n10E	10	5	3617.0	2	100.0	0.8	95.1	99.6	57	0.3	0.5	57	3	1.4
n10f	10	5	3468.0	2	127.1	1.5	74.7	83.6	42	0.4	0.5	148	2613	160.1
n10F	10	5	3601.0	2	102.3	0.5	94.2	96.1	12	0.2	0.4	12	3	1.5
n10g	10	5	4179.0	2	108.1	0.9	77.2	83.1	32	0.3	0.4	90	2011	126.4
n10G	10	5	3763.0	2	129.8	0.6	84.3	88.6	12	0.3	0.4	12	3	0.8
n10h	10	5	3885.0	2	127.6	0.8	89.8	91.8	28	0.3	0.5	30	17	2.2
n10H	10	5	3600.0	2	114.3	1.0	78.9	85.5	34	0.3	0.5	37	89	4.9
n10i	10	5	2548.0	2	123.2	0.5	84.7	93.8	18	0.2	0.3	19	9	2.0
n10I	10	5	3963.0	2	133.9	0.9	82.8	86.3	24	0.3	0.5	25	49	4.4
n10j	10	5	3172.0	2	121.9	0.9	89.6	95.3	26	0.3	0.5	27	15	2.9
n10J	10	5	2992.0	1	100.0	0.5	80.6	100.0	20	0.2	0.4	20	1	0.9
n20A	20	5	4380.0	2	124.4	1.3	87.8	89.9	69	0.5	0.7	466	4431	1513.9
n20B	20	5	4980.0	3	110.1	2.8	86.0	92.8	76	1.0	1.2	177	1245	435.0
n20C	20	5	5453.0	3	122.1	3.4	89.6	91.1	165	0.8	1.1	1565	22505	9362.2
n20D	20	5	5021.0	3	108.6	2.3	91.9	96.4	118	0.9	1.1	127	433	73.8
n20E	20	5	4998.0	2	100.0	1.8	94.6	97.1	64	0.7	0.9	67	23	4.0
n20F	20	5	5031.0	3	124.5	4.6	88.5	93.5	111	1.1	1.4	114	87	16.4
n20G	20	5	5304.0	3	127.9	3.2	83.9	90.9	99	0.9	1.1	151	609	60.6
n20H	20	5	5312.0	2	113.9	2.2	91.5	92.0	41	0.6	0.7	105	143	148.3
n20I	20	5	4793.0	3	128.9	1.9	84.9	89.8	84	0.6	0.8	604	5179	1034.4
n20J	20	5	4139.0	2	103.6	1.2	91.3	98.6	52	0.4	0.6	55	21	5.1
n30A	30	5	6131.0	4	115.2	11.3	92.7	93.8	96	7.5	5.3	372	10261	5259.8
n30B	30	5	7278.0	5	100.0	4.3	74.3	76.5	88	1.4	1.6	959	10923	10800.0
n30C	30	5	6311.0	4	100.0	6.8	86.4	90.4	140	2.3	2.7	1127	12551	10800.0
n30D	30	5	6156.0	4	125.8	7.1	88.3	92.7	173	1.9	2.2	673	16181	10800.0
n30E	30	5	7884.0	4	100.0	5.7	65.9	70.3	149	1.6	1.9	678	4241	10800.0
n30F	30	5	7374.0	4	100.0	7.6	71.1	75.9	181	2.8	2.7	1394	16959	10800.0
n30G	30	5	7273.0	3	109.4	6.9	86.7	95.1	219	2.6	3.1	506	929	1865.2
n30H	30	5	6058.0	4	100.0	5.3	83.6	85.6	137	1.8	2.2	1284	15379	10800.0
n30I	30	5	8452.0	5	100.0	10.1	59.6	61.1	127	5.8	2.0	1223	16985	10800.0
n30J	30	5	7173.0	4	100.0	6.2	74.3	78.9	144	2.0	2.3	1247	14089	10800.0
n40A	40	8	8575.0	5	100.0	12.7	76.8	77.6	215	3.4	3.7	1155	10313	10800.0
n40B	40	8	9883.0	6	100.0	17.9	66.5	67.7	229	3.4	3.6	1221	10081	10800.0
n40C	40	8	6390.0	4	128.2	10.7	94.8	97.9	316	2.4	2.6	362	629	1570.4
n40D	40	8	10531.0	7	100.0	14.3	64.9	66.7	311	2.1	2.2	775	8337	10800.0
n40E	40	8	10573.0	8	100.0	12.6	59.1	62.3	286	2.2	2.5	643	10033	10800.0
n40F	40	8	10398.0	6	100.0	14.9	61.4	62.6	249	2.3	2.4	1077	8603	10800.0
n40G	40	8	10981.0	7	100.0	21.3	64.6	66.5	455	3.9	4.3	921	10263	10800.0
n40H	40	8	9871.0	7	100.0	17.8	66.7	67.6	330	2.9	3.1	1096	8609	10800.0
n40I	40	8	8394.0	5	100.0	15.3	75.9	78.2	319	2.3	2.5	663	6897	10800.0
n40J	40	8	6739.0	4	158.0	18.6	90.9	94.8	330	2.2	2.3	398	431	1441.6

Table EC.5 Detailed results on set S_A with $Q = 20$ and $T = 15$

Name	n	$\|K\|$	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
n10a	10	5	3292.0	1	146.2	0.8	86.3	98.4	34	0.4	0.5	35	17	3.5
n10A	10	5	2471.0	1	146.3	0.3	94.8	99.3	35	0.1	0.2	35	3	3.3
n10b	10	5	2876.0	1	100.0	0.5	86.8	100.0	32	0.2	0.3	32	1	0.8
n10B	10	5	3066.0	1	100.0	0.6	93.1	100.0	44	0.2	0.3	44	1	0.9
n10c	10	5	2948.0	1	100.0	0.1	99.7	99.7	0	0.1	0.1	0	3	0.2
n10C	10	5	2674.0	1	142.9	0.7	85.5	99.5	60	0.3	0.4	60	3	1.8
n10d	10	5	3438.0	1	100.0	0.2	98.3	100.0	24	0.1	0.1	24	1	0.5
n10D	10	5	2885.0	1	134.0	0.6	79.4	98.8	34	0.2	0.2	34	9	2.8
n10e	10	5	2823.0	1	100.0	0.5	90.9	100.0	76	0.2	0.2	76	1	0.8
n10E	10	5	3569.0	1	100.0	0.3	94.4	100.0	56	0.1	0.2	56	1	0.5
n10f	10	5	2996.0	1	163.1	0.4	78.9	94.8	32	0.2	0.2	45	91	10.2
n10F	10	5	3315.0	1	100.0	0.2	94.8	100.0	18	0.1	0.1	18	1	0.4
n10g	10	5	3685.0	1	122.1	0.6	80.2	89.4	30	0.2	0.2	77	851	144.7
n10G	10	5	3115.0	1	100.0	0.3	86.9	100.0	25	0.1	0.1	25	1	0.5
n10h	10	5	3388.0	1	100.0	0.2	97.1	100.0	15	0.1	0.1	15	1	0.4
n10H	10	5	3104.0	1	255.3	0.4	82.6	92.3	34	0.1	0.2	38	119	12.1
n10i	10	5	2279.0	1	100.0	0.3	85.3	100.0	26	0.1	0.1	26	1	0.5
n10I	10	5	3287.0	1	174.7	0.4	89.9	98.0	27	0.2	0.2	29	21	3.7
n10j	10	5	2932.0	1	100.0	0.3	92.6	100.0	28	0.1	0.2	28	1	0.5
n10J	10	5	2992.0	1	100.0	0.2	69.0	100.0	25	0.1	0.1	25	1	0.4
n20A	20	5	4038.0	2	107.9	1.0	85.8	89.4	46	0.5	0.5	590	9653	3839.7
n20B	20	5	4662.0	2	138.5	3.9	80.1	90.8	92	1.1	1.2	1423	13979	10800.0
n20C	20	5	4845.0	2	147.3	5.0	90.8	94.9	156	1.8	2.1	189	1069	955.3
n20D	20	5	4801.0	2	108.1	3.1	88.4	98.5	182	1.2	1.4	185	67	93.5
n20E	20	5	4755.0	2	126.6	2.8	90.7	98.3	159	1.1	1.3	159	3	19.7
n20F	20	5	4620.0	2	153.3	3.5	88.0	93.8	81	1.3	1.4	100	241	134.3
n20G	20	5	4814.0	2	116.5	3.2	82.1	92.6	108	1.2	1.3	186	1973	1084.2
n20H	20	5	4869.0	2	179.4	3.3	87.9	89.7	123	0.5	0.6	201	521	244.4
n20I	20	5	4327.0	2	117.4	3.2	83.9	94.5	119	1.1	1.3	149	345	257.6
n20J	20	5	4098.0	2	112.0	2.2	82.3	92.8	61	1.0	1.3	70	127	25.8
n30A	30	5	5759.0	3	105.8	10.0	85.7	87.3	297	3.8	4.4	1118	10381	10800.0
n30B	30	5	5145.0	2	150.5	11.2	91.2	96.9	228	3.1	3.3	293	891	1515.4
n30C	30	5	6024.0	3	100.0	12.1	79.1	85.3	185	3.5	3.8	470	1061	10800.0
n30D	30	5	6953.0	4	100.0	12.3	66.4	73.2	265	3.0	3.4	523	1559	10800.0
n30E	30	5	5192.0	2	181.4	12.2	86.1	94.3	251	2.4	2.6	1014	6973	10800.0
n30F	30	5	8206.0	4	100.0	15.8	53.9	59.8	261	2.9	3.1	997	11295	10800.0
n30G	30	5	9029.0	4	100.0	12.9	63.9	72.1	384	4.2	4.8	798	3259	10800.0
n30H	30	5	4569.0	2	108.6	6.4	94.0	96.5	214	2.4	2.7	254	173	988.5
n30I	30	5	4847.0	2	126.1	5.3	89.1	92.2	163	1.8	2.0	970	3743	10800.0
n30J	30	5	5041.0	2	115.4	10.9	92.2	98.9	141	3.8	4.1	146	41	159.9
n40A	40	8	8620.0	5	100.0	26.9	65.5	66.8	311	6.9	7.6	1058	6943	10800.0
n40B	40	8	8349.0	5	130.1	33.7	65.8	68.9	248	7.4	7.7	976	5967	10800.0
n40C	40	8	5812.0	3	124.0	27.9	91.1	96.8	431	9.4	10.4	633	5581	10800.0
n40D	40	8	7757.0	5	122.5	36.7	75.1	81.8	325	7.8	8.4	600	5337	10800.0
n40E	40	8	8995.0	4	100.0	24.1	57.8	63.2	231	5.5	5.8	444	7159	10800.0
n40F	40	8	5914.0	3	127.5	28.1	92.2	97.0	273	6.2	6.5	448	1387	2652.4
n40G	40	8	9918.0	5	100.0	55.7	59.0	62.6	528	7.8	8.3	739	4379	10800.0
n40H	40	8	8879.0	5	100.0	40.0	61.4	63.3	309	8.7	9.3	753	5883	10800.0
n40I	40	8	11188.0	5	100.0	34.4	47.8	50.6	464	6.3	7.0	811	8427	10800.0
n40J	40	8	7330.0	4	100.0	37.8	30.7	74.3	194	12.1	13.1	237	1543	10800.0

Table EC.6 Detailed results on set S_B with $Q = 10$ and $T = 3600$

Name	n	$\|K\|$	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
n10a	10	5	3670.0	1	102.3	0.5	88.7	97.0	46	0.2	0.4	46	3	1.1
n10A	10	5	3055.0	2	100.0	0.6	97.1	100.0	36	0.2	0.4	36	1	1.1
n10b	10	5	3538.0	2	111.0	0.9	91.1	97.4	26	0.3	0.5	27	29	3.5
n10B	10	5	4256.0	2	110.9	0.4	89.2	91.4	56	0.1	0.3	110	2161	92.7
n10c	10	5	4408.0	2	100.0	0.3	98.7	100.0	8	0.1	0.3	8	1	0.7
n10C	10	5	3492.0	2	100.0	0.5	91.0	99.1	30	0.2	0.4	39	3	0.8
n10d	10	5	4593.0	2	100.0	0.6	88.5	97.8	27	0.2	0.4	28	35	3.1
n10D	10	5	3273.0	2	147.3	0.8	82.0	97.3	68	0.2	0.4	76	13	1.3
n10e	10	5	4055.0	2	101.5	0.4	93.8	97.7	41	0.1	0.3	46	29	1.3
n10E	10	5	4876.0	2	100.0	0.4	96.3	100.0	81	0.2	0.5	81	1	0.9
n10f	10	5	3468.0	2	106.9	0.6	85.9	90.7	38	0.2	0.4	40	239	10.7
n10F	10	5	4320.0	2	114.7	0.7	89.7	96.0	60	0.2	0.4	67	83	3.2
n10g	10	5	4431.0	2	106.7	0.5	87.1	89.1	14	0.1	0.3	35	423	14.1
n10G	10	5	4790.0	2	104.3	0.4	89.5	92.0	27	0.2	0.3	38	185	5.2
n10h	10	5	4194.0	2	100.0	0.5	97.1	98.7	40	0.1	0.3	41	3	2.9
n10H	10	5	4037.0	2	120.5	0.9	81.4	93.7	63	0.3	0.5	73	167	15.1
n10i	10	5	2677.0	2	100.0	0.4	99.9	100.0	15	0.1	0.3	15	1	0.7
n10I	10	5	3783.0	1	163.5	0.7	90.1	93.1	24	0.3	0.5	24	3	1.3
n10j	10	5	3580.0	2	100.0	0.5	97.3	99.2	54	0.2	0.4	54	3	1.5
n10J	10	5	3604.0	3	100.0	0.7	82.8	93.8	51	0.2	0.4	62	113	4.3
n20A	20	5	5200.0	3	100.0	2.0	96.9	99.5	97	0.8	1.1	97	5	7.1
n20B	20	5	5552.0	3	117.5	3.5	88.0	94.8	100	1.2	1.6	233	1723	325.4
n20C	20	5	6740.0	3	122.7	5.5	93.4	97.2	352	1.7	2.1	476	1339	332.8
n20D	20	5	6745.0	3	100.8	2.3	95.8	97.2	186	0.7	1.2	296	465	82.1
n20E	20	5	6691.0	3	108.1	3.7	90.9	98.4	299	1.4	2.0	366	627	151.6
n20F	20	5	5930.0	4	103.1	3.9	92.9	94.4	102	1.7	1.7	158	373	71.6
n20G	20	5	6070.0	3	108.5	3.5	90.2	96.7	117	1.3	1.6	188	729	130.6
n20H	20	5	6886.0	3	134.6	7.7	89.3	91.3	240	2.2	1.4	522	3671	919.9
n20I	20	5	5724.0	3	113.6	3.9	88.9	94.0	202	1.2	1.3	413	3545	1512.3
n20J	20	5	5290.0	3	151.6	3.3	87.8	91.2	340	0.7	0.9	471	3223	467.0
n30A	30	5	7491.0	4	119.1	40.5	95.7	97.9	344	27.1	8.5	514	2437	1951.7
n30B	30	5	7446.0	4	136.8	18.1	89.4	96.3	575	8.0	4.6	821	1289	704.0
n30C	30	5	9096.0	5	100.0	33.9	77.2	81.3	431	18.0	6.9	1492	13481	10800.0
n30D	30	5	9114.0	5	100.0	29.1	74.3	77.3	654	15.5	6.9	1160	15117	10800.0
n30E	30	5	7454.0	4	101.6	12.7	88.2	94.1	240	6.4	4.7	531	14787	10730.9
n30F	30	5	8541.0	5	100.0	73.6	80.5	84.5	366	53.7	6.8	1308	11799	10800.0
n30G	30	5	10209.0	5	100.0	29.2	88.8	93.4	961	13.8	6.0	2029	11503	10800.0
n30H	30	5	8277.0	4	100.0	26.1	82.1	84.9	408	12.8	7.0	718	12059	10800.0
n30I	30	6	8587.0	6	100.0	10.3	75.5	76.9	197	4.6	3.8	665	17733	10800.0
n30J	30	5	9116.0	5	100.0	52.7	71.7	74.4	507	24.8	5.2	1339	15049	10800.0
n40A	40	8	11640.0	8	100.0	59.7	70.2	74.1	764	24.6	10.3	1402	11369	10800.0
n40B	40	8	10456.0	6	100.0	66.1	70.1	73.1	587	20.3	12.3	1452	7783	10800.0
n40C	40	8	8588.0	5	123.6	32.6	94.6	98.2	520	12.1	8.1	825	11301	10800.0
n40D	40	8	10174.0	5	100.0	43.7	83.1	87.9	581	12.8	12.0	984	5615	10800.0
n40E	40	8	12328.0	8	100.0	41.0	59.4	63.6	783	8.7	7.5	1585	8969	10800.0
n40F	40	8	10972.0	7	100.0	47.9	76.9	80.9	623	19.0	11.3	1529	7269	10800.0
n40G	40	8	12933.0	7	100.0	87.1	67.3	69.3	705	26.0	14.3	1203	6721	10800.0
n40H	40	8	11816.0	7	100.0	110.4	69.1	70.6	708	48.8	12.7	1282	9221	10800.0
n40I	40	8	11072.0	7	100.0	75.0	78.3	83.1	569	30.7	7.8	1413	11667	10800.0
n40J	40	8	10418.0	6	100.0	24.7	70.6	73.9	424	8.0	6.9	1053	9539	10800.0

Table EC.7 Detailed results on set S_B with $Q = 10$ and $T = 7200$

Name	n	$\|K\|$	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
n10a	10	5	3484.0	1	110.2	0.3	90.0	100.0	58	0.7	1.0	58	3	1.4
n10A	10	5	2994.0	1	100.0	0.4	96.2	100.0	28	0.3	0.6	28	1	1.4
n10b	10	5	3122.0	1	100.0	0.4	86.4	100.0	24	0.7	1.0	24	1	1.4
n10B	10	5	3671.0	1	100.0	1.1	96.2	100.0	63	0.5	0.8	63	1	2.1
n10c	10	5	4226.0	1	100.0	0.6	98.0	100.0	16	0.2	0.5	16	1	1.2
n10C	10	5	3380.0	1	122.5	1.6	83.9	97.8	122	0.8	1.2	133	29	16.8
n10d	10	5	4238.0	1	100.0	0.9	89.0	100.0	24	0.4	0.7	24	1	1.8
n10D	10	5	3071.0	1	100.0	1.0	79.3	100.0	80	0.9	1.3	80	1	2.5
n10e	10	5	3816.0	1	100.0	0.1	88.8	100.0	60	0.4	0.7	60	1	0.8
n10E	10	5	4828.0	1	100.0	0.7	90.7	99.9	88	0.6	1.0	88	3	1.6
n10f	10	5	2996.0	1	166.3	2.1	83.9	98.0	65	0.9	1.2	66	13	20.7
n10F	10	5	3758.0	1	100.0	0.9	95.1	100.0	44	0.4	0.7	44	1	1.7
n10g	10	5	3685.0	1	113.4	1.3	87.1	94.2	54	0.5	0.8	58	61	5.0
n10G	10	5	3325.0	1	100.0	0.7	96.0	100.0	12	0.3	0.5	12	1	1.3
n10h	10	5	3960.0	1	117.8	0.5	91.7	99.9	48	0.4	0.7	48	3	1.5
n10H	10	5	3813.0	1	100.8	2.0	79.8	95.1	48	1.5	2.0	69	235	27.2
n10i	10	5	2390.0	1	100.0	0.9	96.5	100.0	25	0.4	0.7	25	1	1.9
n10I	10	5	3287.0	1	250.7	1.6	90.7	97.4	30	0.8	1.1	31	5	3.0
n10j	10	5	3249.0	1	119.9	1.0	98.0	99.1	20	0.5	0.7	20	9	1.9
n10J	10	5	3060.0	1	100.0	0.6	71.6	100.0	31	0.2	0.5	31	1	1.4
n20A	20	5	4815.0	2	111.4	7.6	90.1	97.6	240	2.0	2.9	278	339	113.0
n20B	20	5	4992.0	2	111.2	9.8	83.1	96.5	188	4.7	5.7	194	85	41.1
n20C	20	5	6089.0	2	100.4	5.3	93.8	99.5	699	12.0	14.3	702	5	21.5
n20D	20	5	6149.0	2	105.2	10.0	94.1	99.2	316	3.9	5.1	321	23	31.2
n20E	20	5	6343.0	2	104.1	37.9	88.8	99.1	683	11.1	13.2	692	75	196.6
n20F	20	5	4973.0	2	126.7	9.4	93.7	97.1	407	2.5	3.6	421	141	78.0
n20G	20	5	5364.0	2	139.9	18.6	87.1	97.6	303	5.4	6.4	311	17	110.7
n20H	20	5	5847.0	2	117.2	19.4	87.1	95.5	426	4.9	6.1	533	933	924.3
n20I	20	5	4818.0	2	116.1	15.4	86.6	97.2	386	4.4	5.6	442	283	375.7
n20J	20	5	4402.0	2	134.6	3.7	91.5	96.6	140	1.3	2.0	203	49	121.0
n30A	30	5	8069.0	4	100.0	91.1	73.7	79.0	1559	29.7	33.1	1751	2157	10800.0
n30B	30	5	7645.0	4	100.0	83.8	78.0	84.1	1384	28.6	33.7	1419	519	10800.0
n30C	30	5	8326.0	4	100.0	122.8	70.2	77.1	867	21.3	24.3	1122	1127	10800.0
n30D	30	5	7654.0	3	100.0	108.0	71.5	79.9	973	22.3	25.4	1346	841	10800.0
n30E	30	5	7216.0	3	100.0	53.5	73.3	82.5	534	19.2	21.8	726	1353	10800.0
n30F	30	5	6310.0	3	117.1	29.6	88.4	95.5	398	11.6	13.1	692	4099	10800.0
n30G	30	5	9037.0	2	116.4	111.6	90.4	98.5	1318	41.4	47.1	1764	357	10800.0
n30H	30	5	6194.0	2	119.5	76.4	91.3	96.6	531	28.5	30.3	669	359	10800.0
n30I	30	5	5975.0	3	100.0	43.6	82.4	90.0	551	10.6	12.7	1022	3143	10800.0
n30J	30	5	5984.0	2	103.1	63.4	92.5	98.6	502	17.7	19.9	549	133	3383.8
n40A	40	8	9413.0	5	100.0	134.0	72.4	76.6	1192	29.7	34.2	1375	441	10800.0
n40B	40	8	7295.0	3	100.0	147.7	79.3	85.5	664	43.7	47.3	940	1629	10800.0
n40C	40	8	7901.0	4	100.0	138.5	87.0	93.5	1149	59.8	65.2	1559	1265	10800.0
n40D	40	9	10391.0	4	100.0	358.0	70.2	77.2	1093	117.1	125.7	1159	551	10800.0
n40E	40	8	9677.0	5	100.0	193.7	59.5	68.0	990	42.5	46.4	1361	1591	10800.0
n40F	40	8	9218.0	4	100.0	148.7	73.3	80.0	1101	47.7	51.4	1356	707	10800.0
n40G	40	8	8415.0	4	100.0	198.8	86.2	90.3	876	55.4	57.6	1031	817	10800.0
n40H	40	8	9239.0	5	100.0	124.6	71.0	73.9	1249	26.5	29.8	1436	1539	10800.0
n40I	40	8	9089.0	4	100.0	100.9	74.2	81.3	699	32.5	34.8	1212	761	10800.0
n40J	40	8	10075.0	5	100.0	202.2	57.8	64.3	866	55.8	58.6	1254	1963	10800.0

Table EC.8 Detailed results on set S_C with $|K| = 8$ and $T = 10$

Name	n	Q	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
1Bari30	13	30	13000.0	2	100.0	1.4	92.5	95.8	22	0.6	0.8	27	55	16.3
2Bari20	13	20	13000.0	2	100.0	1.6	92.5	98.4	28	0.8	1.0	28	9	4.3
3Bari10	13	10	14200.0	2	100.0	1.0	95.4	100.0	52	0.3	0.6	52	1	2.2
4ReggioEmilia30	14	30	15800.0	2	107.0	1.2	91.7	92.2	21	0.6	0.8	24	93	13.0
5ReggioEmilia20	14	20	15800.0	2	100.0	1.3	95.3	99.9	61	0.5	0.8	61	1	2.2
6ReggioEmilia10	14	10	20600.0	2	104.9	1.5	93.8	99.0	185	0.4	0.8	190	27	2.6
7Bergamo30	15	30	11300.0	3	100.0	1.5	95.4	100.0	32	0.6	0.8	32	1	2.6
8Bergamo20	15	20	11300.0	3	100.0	0.8	97.8	100.0	44	0.3	0.4	44	1	1.6
9Bergamo12	15	12	11600.0	3	100.0	0.7	98.6	100.0	15	0.1	0.3	15	1	1.1
10Parma30	15	30	24100.0	2	100.0	1.3	98.6	100.0	6	0.6	0.9	6	1	2.3
11Parma20	15	20	24100.0	2	100.0	1.1	98.7	100.0	4	0.6	0.8	4	1	2.2
12Parma10	15	10	24100.0	2	100.0	0.9	100.0	100.0	0	0.3	0.6	0	1	1.5
13Treviso30	18	30	24138.0	3	107.9	1.7	91.5	96.9	22	0.7	0.8	33	99	13.8
14Treviso20	18	20	24138.0	3	107.9	2.4	91.5	96.9	21	0.9	1.2	32	93	16.8
15Treviso10	18	10	24138.0	3	117.8	1.6	92.4	96.9	19	0.6	0.8	37	105	9.5
16LaSpezia30	20	30	19298.0	3	107.5	2.8	92.1	93.5	40	1.0	1.1	205	5441	2148.3
17LaSpezia20	20	20	19298.0	3	107.5	2.5	92.1	93.5	40	0.8	1.0	201	5371	1970.6
18LaSpezia10	20	10	19409.0	3	114.3	2.5	93.1	94.5	55	0.8	1.1	191	3487	539.7
19BuenosAires30	21	30	57020.0	3	103.5	3.9	95.9	98.3	302	0.8	1.2	517	1001	227.6
20BuenosAires20	21	20	64641.0	3	103.0	3.2	97.5	99.1	996	0.5	1.0	2570	585	332.2
21Ottawa30	21	30	17475.0	3	106.5	2.7	97.5	97.5	22	1.2	1.3	22	13	5.7
22Ottawa20	21	20	17475.0	3	106.5	2.4	97.5	97.5	22	0.9	1.1	22	13	5.4
23Ottawa10	21	10	17475.0	3	102.8	1.8	97.5	97.6	29	0.6	0.8	29	13	3.1
24SanAntonio30	23	30	19821.0	3	100.0	3.0	93.3	95.8	82	1.0	1.1	103	369	231.7
25SanAntonio20	23	20	19877.0	3	105.8	1.9	95.5	97.4	98	0.6	0.6	123	97	30.1
26SanAntonio10	23	10	24743.0	4	100.7	2.3	94.5	97.5	371	0.4	0.5	462	1299	449.7
27Brescia30	27	30	24300.0	4	105.3	3.3	93.0	96.9	84	0.8	0.8	551	7989	3210.2
28Brescia20	27	20	24600.0	4	106.1	3.1	92.7	96.1	82	0.6	0.7	931	15875	10800.0
29Brescia11	27	11	25700.0	4	102.3	2.5	94.5	98.1	214	0.4	0.5	510	9283	5546.7
30Roma30	28	30	52400.0	4	144.8	3.4	83.2	99.2	279	0.7	0.8	314	297	366.5
31Roma20	28	20	54200.0	4	100.0	5.0	86.3	100.0	477	1.2	1.7	477	1	7.1
32Roma18	28	18	55100.0	4	101.5	3.1	88.6	99.1	487	0.5	0.7	1335	2201	1574.0
33Madison30	28	30	36204.0	4	105.1	9.6	94.0	96.4	44	2.7	2.7	259	3749	3750.6
34Madison20	28	20	36204.0	4	105.1	13.1	94.0	96.4	44	4.3	4.7	171	2289	2202.6
35Madison10	28	10	38358.0	4	115.8	4.1	92.8	96.9	84	0.9	1.0	298	2189	942.6
36Guadalajara30	41	30	52083.0	8	100.0	59.3	82.3	86.9	53	48.0	20.1	472	2669	10800.0
37Guadalajara20	41	20	49196.0	7	100.0	16.0	87.2	92.0	80	6.0	3.9	191	615	10800.0
38Guadalajara11	41	11	50075.0	6	100.0	19.7	87.8	92.1	352	8.3	7.5	477	573	10800.0
39Dublin30	45	30	37357.0	6	100.0	45.1	84.2	85.9	210	21.4	12.4	1032	7789	10800.0
40Dublin20	45	20	36806.0	6	100.0	38.0	86.9	88.4	233	7.7	8.4	747	5989	10800.0
41Dublin11	45	11	41989.0	7	100.0	34.3	86.4	88.0	1890	6.3	8.4	2408	2367	10800.4
42Denver30	51	30	55212.0	7	100.0	320.7	84.9	89.5	412	160.2	42.2	536	1729	10800.0
43Denver20	51	20	52669.0	7	100.0	167.2	89.9	94.1	321	20.5	21.2	396	319	10800.0
44Denver10	51	10	57707.0	8	100.0	323.3	89.9	96.2	881	225.1	78.4	985	1323	10800.0
45RioDeJaneiro30	55	30	136797.0	7	100.0	1864.2	91.3	92.1	331	1641.7	508.8	718	3171	10800.0
46RioDeJaneiro20	55	20	145392.0	8	100.0	1594.6	87.2	88.0	1493	1312.4	290.2	1831	4349	10800.0
47RioDeJaneiro10	55	10	162198.0	8	100.0	690.4	94.3	97.0	9328	414.3	150.5	9546	227	10800.0

Table EC.9 Detailed results on set S_C with $|K| = 8$ and $T = 15$

Name	n	Q	z^*	rt	$\%UB$	t_{UB}	$\%LB$	$\%LB_r$	cut_r	t_{PA}	t_r	cut	$node$	t
1Bari30	13	30	12000.0	1	100.0	2.2	94.7	100.0	12	1.3	1.4	12	1	3.7
2Bari20	13	20	12700.0	1	100.0	0.9	90.8	100.0	24	0.5	0.5	24	1	1.5
3Bari10	13	10	14200.0	2	100.0	0.6	95.1	100.0	57	0.1	0.1	57	1	1.1
4ReggioEmilia30	14	30	13100.0	1	100.0	0.9	100.0	100.0	5	0.4	0.4	5	1	1.4
5ReggioEmilia20	14	20	15800.0	2	131.6	0.6	91.6	99.6	81	0.2	0.2	81	9	2.5
6ReggioEmilia10	14	10	20300.0	2	100.0	0.8	94.9	100.0	135	0.3	0.4	135	1	1.3
7Bergamo30	15	30	11300.0	2	100.0	1.1	94.5	100.0	42	0.2	0.2	42	1	1.6
8Bergamo20	15	20	11300.0	3	100.0	0.4	97.3	100.0	38	0.1	0.1	38	1	0.9
9Bergamo12	15	12	11600.0	3	100.0	0.3	98.6	100.0	21	0.0	0.1	21	1	0.7
10Parma30	15	30	24100.0	2	100.0	1.0	97.2	100.0	16	0.3	0.3	16	1	1.4
11Parma20	15	20	24100.0	2	100.0	0.7	97.7	100.0	16	0.2	0.2	16	1	0.9
12Parma10	15	10	24100.0	2	100.0	0.2	100.0	100.0	1	0.0	0.1	1	1	0.3
13Treviso30	18	30	22837.0	2	100.0	4.5	91.2	100.0	31	1.4	1.5	31	1	1.5
14Treviso20	18	20	22837.0	2	100.0	2.6	91.2	100.0	27	1.1	1.2	27	1	3.8
15Treviso10	18	10	22837.0	2	100.0	0.8	93.1	100.0	24	0.3	0.4	24	1	1.3
16LaSpezia30	20	30	17882.0	2	107.1	7.8	91.1	93.7	38	3.9	4.1	622	10583	10800.0
17LaSpezia20	20	20	17874.0	2	107.1	4.3	91.2	93.7	42	1.7	1.8	644	11873	10800.0
18LaSpezia10	20	10	17863.0	2	116.9	2.2	94.3	96.6	58	0.9	1.0	100	809	443.3
19BuenosAires30	21	30	56898.0	3	105.4	2.9	95.2	98.0	309	0.6	0.7	937	3737	2720.6
20BuenosAires20	21	20	64087.0	3	100.0	1.9	98.2	99.9	577	0.5	0.8	598	5	4.1
21Ottawa30	21	30	16198.0	2	107.9	6.7	93.8	94.3	32	2.2	2.3	116	617	702.2
22Ottawa20	21	20	16198.0	2	109.3	5.0	93.8	94.3	32	1.7	1.7	105	487	491.2
23Ottawa10	21	10	16196.0	2	100.0	2.6	94.2	95.9	67	1.0	1.1	88	187	148.2
24SanAntonio30	23	30	17279.0	2	100.0	10.0	92.9	99.2	139	4.1	4.3	140	9	29.7
25SanAntonio20	23	20	17581.0	2	100.0	3.9	97.1	100.0	151	1.5	1.6	151	1	5.8
26SanAntonio10	23	10	23893.0	3	100.0	3.5	97.0	99.9	370	0.9	1.0	370	1	4.7
27Brescia30	27	30	22900.0	3	112.7	11.2	93.3	98.4	78	3.0	3.1	199	491	3061.8
28Brescia20	27	20	23300.0	3	105.2	9.0	93.2	97.6	104	3.3	3.8	343	6221	6753.2
29Brescia11	27	11	25100.0	3	102.0	5.1	93.8	99.0	293	0.9	1.1	471	2813	2535.5
30Roma30	28	30	51800.0	3	106.4	8.5	77.6	99.6	278	2.3	2.7	305	125	464.8
31Roma20	28	20	54200.0	4	100.0	5.3	84.4	100.0	371	1.1	1.4	371	1	7.1
32Roma18	28	18	54800.0	3	103.5	6.3	87.2	99.4	532	0.7	0.9	1627	4741	10800.0
33Madison30	28	30	32407.0	3	107.8	27.4	92.3	95.3	92	8.8	8.9	347	1967	10800.0
34Madison20	28	20	32407.0	3	107.6	15.8	92.3	95.4	91	5.2	5.3	339	2053	7695.5
35Madison10	28	10	34815.0	2	122.7	8.6	90.6	95.6	103	2.5	2.7	525	2197	3579.2
36Guadalajara30	41	30	43867.0	4	101.1	39.9	90.5	96.9	59	14.2	14.2	199	1563	10800.0
37Guadalajara20	41	20	43963.0	4	100.9	30.0	90.8	96.7	70	9.4	9.5	408	2987	10800.0
38Guadalajara11	41	11	44411.0	4	108.5	34.7	93.2	98.6	738	7.3	8.8	856	451	3364.2
39Dublin30	45	30	36032.0	5	100.0	74.2	75.6	77.5	259	10.7	10.9	687	3721	10800.0
40Dublin20	45	20	36841.0	5	100.0	46.0	77.2	79.4	661	7.9	8.7	994	4509	10800.0
41Dublin11	45	11	39408.0	5	100.0	63.5	87.8	90.6	2055	8.8	13.6	2960	1871	10800.0
42Denver30	51	30	50270.0	5	100.0	193.2	84.6	91.6	340	37.8	39.0	426	499	10800.0
43Denver20	51	20	48122.0	5	100.0	127.4	90.5	96.7	486	24.0	25.0	687	1553	10800.0
44Denver10	51	10	53320.0	7	100.0	61.4	90.9	99.8	2276	13.1	17.3	2714	541	10800.0
45RioDeJaneiro30	55	30	119839.0	5	100.0	608.8	85.6	87.6	618	88.7	90.6	751	1013	10800.0
46RioDeJaneiro20	55	20	124209.0	5	100.0	345.4	86.7	89.0	2256	61.3	66.8	2632	1483	10800.0
47RioDeJaneiro10	55	10	156826.0	6	100.0	1495.8	96.1	99.6	15515	191.8	313.7	15528	79	10800.0

Table EC.10 Detailed results of the pricing algorithms on set S_A

				EXM pricing algorithm						Pricing algorithm based on Casazza et al. (2019)					
Name	n	$\|K\|$	LB	t_{LP}	t_{PA}	fw	fwd	bw	bwd	t_{LP}	t_{PA}	fw	fwd	bw	bwd
n10a	10	5	3210.46	0.22	0.12	1295	7699	1153	7182	0.25	0.12	10116	179826	3841	68628
n10A	10	5	2920.76	0.07	0.06	1289	7886	800	4615	0.09	0.08	13319	248455	5481	106743
n10b	10	5	2687.58	0.06	0.05	1611	10768	708	4274	0.06	0.05	7421	95630	4787	60123
n10B	10	5	3540.84	0.04	0.03	1096	6971	862	5552	0.08	0.07	13907	295130	6385	130118
n10c	10	5	4183.67	0.04	0.04	1153	7460	716	4586	0.06	0.05	9525	181410	3767	67273
n10C	10	5	2878.65	0.04	0.04	1242	7641	895	5536	0.07	0.06	13383	276098	5247	103073
n10d	10	5	3897.11	0.04	0.03	994	5904	837	5400	0.05	0.04	9599	165563	4193	67251
n10D	10	5	2609.49	0.03	0.03	862	5677	533	3309	0.07	0.06	13227	252539	5442	99873
n10e	10	5	3403.77	0.04	0.03	1300	7867	847	4962	0.07	0.06	13018	279380	5027	114377
n10E	10	5	4489.26	0.03	0.02	1253	7763	829	5005	0.09	0.08	13725	266647	5237	108301
n10f	10	5	2656.90	0.04	0.04	1380	8916	1154	7203	0.04	0.04	10434	147948	4935	63680
n10F	10	5	3662.62	0.05	0.04	1055	6473	1046	6358	0.08	0.07	21457	418085	8588	157857
n10g	10	5	3417.47	0.04	0.04	1336	8495	1026	6297	0.03	0.03	10652	148853	4986	66678
n10G	10	5	3435.92	0.05	0.04	1588	10325	1082	6433	0.06	0.05	14029	255705	8463	145785
n10h	10	5	3683.04	0.02	0.02	728	4769	571	3674	0.06	0.04	11629	256757	5990	129142
n10H	10	5	3196.66	0.04	0.04	1797	12068	922	5400	0.06	0.05	12486	210277	6164	93828
n10i	10	5	2424.54	0.03	0.03	1128	7155	843	5213	0.05	0.04	11756	248329	5727	114724
n10I	10	5	3325.31	0.04	0.04	1423	8685	936	5687	0.06	0.05	15879	270575	8381	120506
n10j	10	5	3247.21	0.05	0.03	1422	8486	1350	8644	0.04	0.03	11742	197631	5351	79767
n10J	10	5	2481.71	0.02	0.02	1032	5650	715	3775	0.06	0.06	14687	276466	7171	132925
n20A	20	5	4536.44	0.08	0.08	2517	30782	1929	22382	0.19	0.17	44854	1484500	13808	454721
n20B	20	5	4531.57	0.13	0.10	3702	48873	2588	31230	0.19	0.17	35617	1029506	20913	595797
n20C	20	5	5984.86	0.14	0.13	3813	56161	2887	41968	0.29	0.26	56568	2245901	17478	671402
n20D	20	5	6003.32	0.10	0.09	2623	36310	2003	27690	0.23	0.21	54882	2262819	16502	637700
n20E	20	5	5753.30	0.08	0.07	2668	33141	1886	23269	0.31	0.29	70617	2881158	17137	702416
n20F	20	5	4839.56	0.18	0.17	4108	59754	3143	45840	0.27	0.25	47092	1588739	16122	531393
n20G	20	5	4982.84	0.11	0.10	3753	51035	2844	37447	0.31	0.30	53745	1866876	20101	679978
n20H	20	5	5325.79	0.12	0.11	3973	52095	2260	27363	0.23	0.21	43700	1563580	19655	666687
n20I	20	5	4440.42	0.11	0.10	3398	45527	2909	37963	0.28	0.26	57772	2295937	22516	810477
n20J	20	5	4184.48	0.05	0.05	2273	26140	1698	18294	0.16	0.15	39773	1245442	14243	418189
n30A	30	5	6349.58	1.71	1.62	56385	1426499	6406	134728	3.53	3.13	674153	30257204	67189	3300935
n30B	30	5	6330.89	0.27	0.22	7745	146530	4611	82087	0.89	0.80	130008	5974514	38371	1680991
n30C	30	5	6286.25	0.37	0.35	10154	218637	6581	132497	1.56	1.49	240993	12669772	62448	3285250
n30D	30	5	6205.94	0.36	0.33	10744	239889	5613	116114	1.29	1.23	173304	9847364	60720	2959403
n30E	30	5	5904.85	0.26	0.24	8251	160423	6124	112176	1.10	1.06	163593	8165108	55175	2530636
n30F	30	5	5934.98	0.48	0.45	14419	337181	6904	143155	1.33	1.26	194338	9641139	63077	2903879
n30G	30	5	8583.22	0.29	0.27	9805	209045	5063	99174	1.57	1.47	229395	12397868	52668	2784856
n30H	30	5	6144.99	0.30	0.28	10497	224716	6127	117241	1.14	1.09	169089	8154504	54269	2490230
n30I	30	5	5463.43	0.30	0.27	10430	219272	6274	117921	1.11	1.06	176453	9357450	58018	2964312
n30J	30	5	6188.78	0.31	0.29	10456	223443	6168	119002	0.82	0.77	148975	6980344	50911	2440887
n40A	40	8	7461.64	0.68	0.65	17652	527935	12711	358630	3.80	3.49	366716	27379934	116736	7955340
n40B	40	8	6776.35	0.74	0.71	20899	598727	14351	386624	2.76	2.63	279112	17943842	107500	6347114
n40C	40	8	7406.27	0.51	0.48	14760	399686	9593	248351	2.35	2.23	264213	18144923	70780	4709741
n40D	40	8	7957.55	0.54	0.51	14140	395622	10712	281881	2.11	1.97	237273	16154355	81492	5157916
n40E	40	8	6633.23	0.25	0.24	8010	208214	5212	125564	1.62	1.54	207328	13982110	81558	4851519
n40F	40	8	7375.05	0.65	0.62	16476	477856	12356	329648	2.30	2.19	272667	18757723	85861	5394279
n40G	40	8	8118.96	0.66	0.64	17376	531241	12127	345382	2.23	2.13	257001	16443458	86367	5484495
n40H	40	8	7386.82	0.69	0.66	18667	570597	11026	306135	2.35	2.23	268078	17299706	101216	6273152
n40I	40	8	7417.07	0.50	0.47	12483	360401	8509	223161	2.17	2.08	281657	19028251	77194	4929908
n40J	40	8	6842.06	0.45	0.43	13259	374194	8015	212394	1.94	1.85	261514	16575702	80910	5112481

Table EC.11 Detailed results of the pricing algorithms on set S_A-3

				EXM pricing algorithm						Pricing algorithm based on Casazza et al. (2019)					
Name	n	$\|K\|$	LB	t_{LP}	t_{PA}	fw	fwd	bw	bwd	t_{LP}	t_{PA}	fw	fwd	bw	bwd
n10a	10	5	3210.46	0.02	0.02	1199	7386	990	6133	0.66	0.51	75532	4059582	24419	1256555
n10A	10	5	2920.76	0.02	0.01	1060	6708	648	3856	0.43	0.41	81000	4058653	30545	1897964
n10b	10	5	2687.58	0.02	0.02	1448	9340	733	4431	0.24	0.22	44033	1678240	26549	984305
n10B	10	5	3540.84	0.02	0.01	970	6308	702	4417	0.48	0.47	95236	5825666	41190	2586838
n10c	10	5	4183.67	0.02	0.01	967	6532	639	4027	0.47	0.45	93910	5146221	20951	1356481
n10C	10	5	2878.65	0.01	0.01	905	5602	623	4032	0.53	0.51	90407	5247109	34566	1975468
n10d	10	5	3897.11	0.02	0.01	1061	6439	920	6052	0.38	0.37	77417	3945299	13157	710748
n10D	10	5	2609.49	0.02	0.01	1082	6863	745	4552	0.47	0.46	102084	5496797	31631	1716018
n10e	10	5	3403.77	0.02	0.02	1211	7523	815	4822	0.45	0.44	79163	4648446	27979	1912797
n10E	10	5	4489.26	0.02	0.01	1149	6955	736	4398	0.38	0.37	82734	4526282	25140	1465485
n10f	10	5	2656.90	0.03	0.02	1632	10352	1429	8673	0.35	0.34	88298	3493301	28483	1038123
n10F	10	5	3662.62	0.02	0.02	1151	7080	1185	7314	0.72	0.71	138380	8332747	46622	2396069
n10g	10	5	3417.47	0.02	0.02	1301	8050	831	5246	0.26	0.23	72538	2929255	31682	1252324
n10G	10	5	3435.92	0.02	0.02	1400	9066	981	5769	0.45	0.44	95400	4584860	48731	2340879
n10h	10	5	3683.04	0.01	0.01	818	5200	626	4048	0.55	0.54	94824	6436342	43833	2825061
n10H	10	5	3196.66	0.02	0.02	1881	12641	936	5511	0.33	0.32	87921	3880680	45893	2026163
n10i	10	5	2424.54	0.02	0.01	1183	7595	864	5261	0.45	0.44	87300	5057647	39376	2090985
n10I	10	5	3325.31	0.02	0.02	1603	10212	979	6196	0.52	0.51	123629	6148698	49886	1994902
n10j	10	5	3247.21	0.02	0.02	1321	8176	1345	8720	0.26	0.25	73333	3599945	23962	1053735
n10J	10	5	2481.71	0.01	0.01	796	4306	562	3021	0.44	0.43	107725	5591096	44922	2548576
n20A	20	5	4536.44	0.06	0.05	2737	32882	2145	25037	3.32	3.28	443912	43121024	124897	11695197
n20B	20	5	4531.57	0.09	0.08	3922	51239	2709	32940	2.64	2.58	307507	25273854	168025	13241426
n20C	20	5	5984.86	0.13	0.12	3989	58026	3020	43086	4.95	4.87	565810	63771825	114221	13001572
n20D	20	5	6003.32	0.07	0.06	2689	37956	2163	29695	5.92	5.84	578622	69047130	135188	14708974
n20E	20	5	5753.30	0.07	0.06	2923	36847	1903	23505	5.21	5.15	686398	83361354	125350	14754613
n20F	20	5	4839.56	0.10	0.09	4155	59390	3658	52231	3.81	3.76	443544	44722604	152581	14203382
n20G	20	5	4982.84	0.08	0.07	4047	54324	3217	42035	4.17	4.13	536860	52707442	177177	17278303
n20H	20	5	5325.79	0.07	0.06	3532	45894	2072	25121	3.24	3.19	374961	38549588	156528	14992064
n20I	20	5	4440.42	0.07	0.07	3328	44704	2607	34185	5.14	5.10	572084	68727871	201839	20558300
n20J	20	5	4184.48	0.04	0.04	2143	24359	1489	16151	2.07	2.04	361871	32684808	97298	7832545
n30A	30	5	6349.58	1.19	1.13	46855	1179426	6052	132026	42.08	41.11	7518902	964230906	439074	62481993
n30B	30	5	6330.89	0.21	0.18	7002	132197	4389	78259	8.41	8.26	1469045	182658863	319257	38852489
n30C	30	5	6286.25	0.32	0.29	9675	212354	5699	114102	15.29	15.15	2355257	355641230	498665	75243238
n30D	30	5	6205.94	0.32	0.30	10316	229184	5475	114658	11.86	11.74	1607706	269818564	515406	71030674
n30E	30	5	5904.85	0.25	0.23	7987	156519	5355	97746	10.13	10.05	1596974	235981030	431013	57006355
n30F	30	5	5934.98	0.46	0.43	13708	320136	6365	133788	11.68	11.56	1701316	258495307	511701	67573123
n30G	30	5	8583.22	0.29	0.27	9517	202617	5031	98396	12.72	12.59	1932139	300326473	371877	56362814
n30H	30	5	6144.99	0.32	0.30	10801	230428	5794	113199	9.71	9.62	1547122	217369106	459550	59098578
n30I	30	5	5463.43	0.30	0.27	10299	217349	6051	113593	11.37	11.26	1685890	254942114	486980	71570948
n30J	30	5	6188.78	0.29	0.27	9814	207544	6262	120676	8.62	8.53	1421050	188666310	404595	52421910
n40A	40	8	7461.64	0.73	0.70	17656	533450	12360	348809	32.09	31.90	3514200	782776318	1063877	206802523
n40B	40	8	6776.35	0.73	0.70	19495	563717	13126	351033	22.18	21.98	2601061	489332814	908405	146955703
n40C	40	8	7406.27	0.59	0.56	16273	434869	11233	292509	21.63	21.43	2585768	499430889	597004	113821299
n40D	40	8	7957.55	0.60	0.57	15459	428859	11699	316665	20.91	20.72	2387709	485151456	659014	117801454
n40E	40	8	6633.23	0.32	0.30	9851	252473	6772	163729	17.82	17.70	2119375	413324121	746104	125654665
n40F	40	8	7375.05	0.68	0.65	16628	482509	12205	329692	25.17	24.96	2940877	596251755	835060	149749272
n40G	40	8	8118.96	0.72	0.69	18442	554633	13638	386288	22.20	21.99	2557231	483219065	691324	128098557
n40H	40	8	7386.82	0.66	0.64	18278	555759	11512	324029	20.42	20.23	2255555	430892215	803073	141456285
n40I	40	8	7417.07	0.56	0.53	13459	385454	8880	235315	23.79	23.61	2809435	568052227	795548	142507998
n40J	40	8	6842.06	0.54	0.51	14167	397045	9328	245420	20.76	20.60	2683575	490000582	718190	128810720

Table EC.12 Detailed results of the pricing algorithms on set S_A-6

				EXM pricing algorithm						Pricing algorithm based on Casazza et al. (2019)					
Name	n	$\|K\|$	LB	t_{LP}	t_{PA}	fw	fwd	bw	bwd	t_{LP}	t_{PA}	fw	fwd	bw	bwd
n10a	10	5	3210.46	0.02	0.02	1199	7386	990	6133	2.09	1.98	371406	40542274	124737	11888301
n10A	10	5	2920.76	0.02	0.01	1066	6756	673	4020	2.11	2.08	467643	44498879	145279	17153340
n10b	10	5	2687.58	0.02	0.02	1448	9340	733	4431	0.67	0.66	180840	12666001	95102	7162670
n10B	10	5	3540.84	0.01	0.01	996	6294	763	4750	2.21	2.19	517395	61256157	156652	19937020
n10c	10	5	4183.67	0.01	0.01	866	5862	524	3300	1.20	1.19	341024	34911924	51133	6851819
n10C	10	5	2878.65	0.01	0.01	905	5602	623	4032	1.88	1.87	414631	47390098	169011	18211082
n10d	10	5	3897.11	0.02	0.01	1061	6439	920	6052	1.32	1.31	413892	43614881	58129	6156092
n10D	10	5	2609.49	0.02	0.01	1082	6863	745	4552	2.11	2.08	508548	57077431	156086	17024103
n10e	10	5	3403.77	0.02	0.02	1348	8292	911	5344	1.74	1.73	383098	43824152	114381	16366554
n10E	10	5	4489.26	0.02	0.02	1149	6955	736	4398	1.14	1.13	253239	28461209	96761	11047408
n10f	10	5	2656.90	0.03	0.02	1632	10352	1429	8673	1.17	1.16	362258	26730911	97217	6648712
n10F	10	5	3662.62	0.02	0.02	1157	7118	1179	7320	2.93	2.92	676891	81624506	186770	19161461
n10g	10	5	3417.47	0.02	0.02	1301	8050	831	5246	1.09	1.08	292394	23491734	137540	10819490
n10G	10	5	3435.92	0.02	0.02	1400	9066	981	5769	1.61	1.60	332010	32062394	190918	17742455
n10h	10	5	3683.04	0.01	0.01	802	5235	647	4135	2.47	2.46	475882	63798680	213089	26542433
n10H	10	5	3196.66	0.03	0.02	2080	13673	1022	5932	1.26	1.25	306020	26624320	156276	14065420
n10i	10	5	2424.54	0.02	0.01	1183	7595	864	5261	1.61	1.60	317851	37815773	180212	18832782
n10I	10	5	3325.31	0.02	0.02	1586	10256	991	6256	2.50	2.49	653950	61982250	250065	19032795
n10j	10	5	3247.21	0.03	0.02	1614	9846	1813	11632	1.26	1.25	358598	34861534	83481	7079544
n10J	10	5	2481.71	0.01	0.01	892	4840	622	3318	1.77	1.76	457866	43501282	162791	17378181
n20A	20	5	4536.44	0.06	0.05	2511	30413	1954	23317	13.63	13.58	1954281	374413216	736444	129093002
n20B	20	5	4531.57	0.07	0.06	3241	42671	2067	25137	9.07	9.03	1206584	194409604	754394	111768439
n20C	20	5	5984.86	0.10	0.09	3907	56255	3000	42553	17.79	17.75	2054466	512758945	666782	139793697
n20D	20	5	6003.32	0.09	0.08	2943	41268	2299	31961	17.30	17.25	2218089	536815704	689631	137620298
n20E	20	5	5753.30	0.07	0.06	2923	36847	1903	23505	19.81	19.76	2525020	632405300	624334	134528299
n20F	20	5	4839.56	0.11	0.10	4008	56673	3417	48990	14.56	14.49	1628667	335509028	1011548	179806680
n20G	20	5	4982.84	0.10	0.09	4375	58953	3524	46335	16.65	16.61	2307276	442336320	818004	155477742
n20H	20	5	5325.79	0.07	0.07	3532	46100	2114	25506	16.59	16.52	2057291	426082551	1007072	182476983
n20I	20	5	4440.42	0.08	0.07	3585	48107	2852	37032	22.34	22.29	2713854	645890027	925080	181013535
n20J	20	5	4184.48	0.05	0.04	2278	25827	1691	18112	9.93	9.90	1547675	282368250	530279	77564151
n30A	30	5	6349.58	1.17	1.11	45430	1139947	7095	151918	98.73	98.44	8775286	2272313750	4664670	1029955309
n30B	30	5	6330.89	0.24	0.22	8440	160136	5035	89859	45.85	45.71	4021344	1050025366	2563835	564196813
n30C	30	5	6286.25	0.33	0.31	9768	212194	5981	119887	84.03	83.83	6736015	2022433178	3545134	1003158766
n30D	30	5	6205.94	0.34	0.32	10535	232631	5964	124916	72.10	71.95	5563536	1840029345	3041073	751638437
n30E	30	5	5904.85	0.22	0.21	7715	152440	4992	91180	76.64	76.48	6395636	1921725290	3352924	847497844
n30F	30	5	5934.98	0.44	0.42	13761	326110	6094	124540	82.43	82.27	7104422	2196317697	2606171	648366230
n30G	30	5	8583.22	0.33	0.31	10614	226142	5563	108166	75.35	75.10	5600324	1741729741	3765603	1000866719
n30H	30	5	6144.99	0.34	0.32	11630	247634	6935	134016	72.54	72.39	6572720	1874637674	3085570	751862301
n30I	30	5	5463.43	0.29	0.27	9882	208117	6048	114880	68.41	68.25	4922154	1556487024	3509424	940729474
n30J	30	5	6188.78	0.34	0.32	11339	241498	6345	126379	50.12	49.89	4064188	947823047	3428706	751084777
n40A	40	8	7461.64	0.70	0.67	17626	534110	12170	343819	2259.45	2256.01	151783977	65652120033	28085178	8335412613
n40B	40	8	6776.35	0.70	0.67	19468	562248	12771	343776	201.45	200.96	9906798	3762100065	9402233	2661028329
n40C	40	8	7406.27	0.56	0.53	15648	417915	11141	280670	152.75	152.39	9667727	3680229149	3863903	1161050742
n40D	40	8	7957.55	0.56	0.54	14915	412596	11448	306827	694.42	692.56	53004521	19535187422	5647654	1577969886
n40E	40	8	6633.23	0.30	0.28	9542	239527	7119	172308	153.90	153.57	9395190	3254992231	5962875	1709081626
n40F	40	8	7375.05	0.64	0.62	15792	454018	12107	330120	200.36	200.03	10418687	4282175711	6837823	2350569860
n40G	40	8	8118.96	0.83	0.79	20546	624980	14244	404436	132.15	131.78	9233043	3342745139	2592672	716269179
n40H	40	8	7386.82	0.77	0.74	20219	614727	12984	362466	133.51	133.22	6664657	2480670388	5642952	1791161737
n40I	40	8	7417.07	0.60	0.57	14503	413247	10510	278373	410.15	409.56	31211706	11855352728	4214124	1268010326
n40J	40	8	6842.06	0.48	0.46	14211	396870	9215	241618	137.52	137.22	7368882	2836579014	5506670	1807842865