Erratum to: A Branch-and-Price algorithm for two multi-compartment vehicle routing problems

Samira Mirzaei¹ · Sanne Wøhlk¹

Published online: 21 April 2017

© Springer-Verlag Berlin Heidelberg and EURO - The Association of European Operational Research Societies 2017

Abstract Despite the vast body of literature on vehicle routing problems, little attention has been paid to multi-compartment vehicle routing problems that investigate transportation of different commodities on the same vehicle, but in different compartments. In this project, we present two strategically different versions of the MCVRP in general settings. In the first version, different commodities may be delivered to the customer by different vehicles, but the full amount of each product must be delivered by a single vehicle. In the second version, each customer may only be serviced by a single vehicle, which must deliver the full amount of all commodities demanded by that customer. We present a Branch-and-Price algorithm for solving the two versions of the problem to optimality and we analyze the effect of the strategic decision of whether or not to allow multiple visits to the same customer by comparing the optimal costs of the two versions. Computational results are presented for instances with up to 100 customers and the algorithm can solve instances with up to 50 customers and 4 commodities to optimality.

After the publication of the article the authors found an erratic value in the code that was used to determine the optimal solutions. Since the error was continuous, we have decided to bring the entire article in its corrected form on the following pages. Please excuse this mistake.

The publisher

The online version of the original article can be found under doi:[10.1007/s13676-016-0096-x](https://doi.org/10.1007/s13676-016-0096-x).

✉ Sanne Wøhlk
sanw@econ.au.dk

¹ Cluster for Operations Research and Logistics (CORAL), Department of Economics and Business Economics, Aarhus University, Aarhus, Denmark

1 Introduction

Vehicle routing problems have been the object of numerous studies and a very large number of papers propose solution methods for solving these problems, see Laporte et al. (2000) and Toth and Vigo (2014). Yet, despite the progress in the field, many challenges still exist and new ones are emerging. In practice, for petroleum and animal feeds distribution, waste collection, home delivery of groceries, etc. the distribution systems are often challenged by a situation where the customers request delivery of different commodities, and these commodities need to be kept segregated during transport. For bulk material, such separation into compartments is necessary to avoid mixing different types of material. For home delivery of groceries, the need for multiple compartments originates from the different temperature requirements for different products. When different types of waste are co-collected, it is important for further treatment that the waste is kept separated during transport. Models and algorithms for solving such multi-compartment problems are thus highly relevant in practice.

In the academic literature, there is, however, a tendency to simplify problems by considering single-commodity models. Most studies assume that each customer requires only one type of commodity, whereas few papers in this area investigate multi-compartment vehicle routing problems (MCVRP). Furthermore, most of the papers that do consider multiple compartments, apply a heuristic based solution approach.

In this project, we present two strategically different versions of the MCVRP. In the first version, which we refer to as Commodity-Split (hereafter C-Split) MCVRP, a customer may receive different commodities from different vehicles. However, the full amount of each commodity must still be delivered by a single vehicle, i.e. we do not allow split delivery as regards the individual commodity. In the second version, which we refer to as No-Split MCVRP, each customer may only be serviced by a single vehicle, which must deliver the full amount of all commodities demanded by that customer. From a customer perspective, a No-Split service will often be perceived as the best service. This is particularly the case for private citizens who prefer that all ordered groceries arrive in one delivery and that all waste is collected jointly by a single vehicle.

This project has two purposes. Firstly, we develop a Branch-and-Price algorithm for solving the two versions of the MCVRP to optimality. Our algorithm uses heuristic as well as exact column generation and is able to solve instances with up to 50 nodes and 4 commodities to optimality within an hour. Not all instances up to this size could be solved, however. Secondly, we analyze the effect of the strategic decision of whether or not to allow deliveries to a single customer to be shared among several vehicles by comparing the costs of the C-Split and the No-Split versions of the MCVRP.

2 Literature review

The most important application of MCVRP is related to petroleum distribution and is known as the petrol station replenishment problem. Brown and Graves (1981) consider a petroleum products dispatching problem from a single terminal to a set of customers. They use optimization routines for automation of this problem and in order to reduce manual operations and operating costs. In addition to optimization algorithms, they propose a heuristic sequential network assignment for this problem. Brown et al. (1987) explain the computer system of Mobil Oil Corporation for centralized control of petroleum distribution to their customers in the United States. This system exploits the knowledge of humans and assists Mobil to reduce its costs and staff and improve the customer service.

Van der Bruggen et al. (1995) redesign the distribution structure of a large oil company in the Netherlands. They choose a hierarchical approach and decompose the problem into three sub-problems including assignment of clients to depots, assignment of products to truck compartments, and route planning. They apply a heuristic for solving the VRP sub-problem. Ben Abdelaziz et al. (2002) solve the petroleum delivery problem of a Tunisian company. They present a mathematical model for this problem. They design a set of least-cost vehicle routes subject to several constraints and they apply the Variable Neighborhood Search heuristic to reach a near-optimal solution.

Avella et al. (2004) investigate the problem of delivering different types of fuel to a set of pumps. They formulate this problem as a set partitioning model and solve it by a Branch-and-Price algorithm. The initial columns for the Branch-and-Price algorithm are provided by an innovative combinatorial heuristic. Cornillier et al. (2008a) consider delivery of petroleum products to petrol stations and decompose this problem into a truck loading problem and a routing problem. They first use a heuristic for solving the loading problem. Due to the structure of the problem, there are three possible outcomes: (1) an optimality test proves that the resulting solution is optimal; (2) a solution was produced but the optimality test failed; and (3) The heuristic did not obtain a feasible solution. In the latter two cases, the algorithm proceeds by solving an ILP model to obtain an optimal solution. They use two different algorithms, based on a matching approach or a column generation scheme for solving the routing sub-problem. Cornillier et al. (2008b) investigated the same problem in a multi-period setting. They present a mathematical model and a heuristic for this problem. The heuristic includes procedures for route construction, truck loading, route packing, and postponement of deliveries. Inspired by the same type of problem, Coelho and Laporte (2015) study a multi-period inventory routing problem. They present a classification of this problem, which, on one hand, considers the option of allowing the content of each vehicle compartment to be split between multiple customers (or multiple tanks at the same customer), and on the other hand, considers the option of allowing the delivery to each customer tank to come from multiple vehicles. They present an exact algorithm for solving the four types of problems originating from this classification. They also consider the problems in a single period setting, which, due to the splitting definition, are

variations of multi-commodity VRP problems that differ from the ones studied in the present paper.

Distribution of animal feed is another application of MCVRP. El Fallahi et al. (2008) consider a fleet of multi-compartment vehicles where each compartment is assigned to one commodity and the full demand of each customer for a commodity must be delivered by a single vehicle. It is, however, possible to deliver different commodities by different vehicles to the same customer. This is what we refer to as the C-Split MCVRP in this paper. They present a mathematical model for this problem and propose a heuristic approach based on a Memetic algorithm and Tabu search for solving it.

Due to the increased focus on sorting, waste collection is a more recent application of MCVRP. Muyldermans and Pang (2010) investigate this problem with assumptions similar to El Fallahi et al. (2008). They propose a local search procedure using 2-opt, cross, exchange, and relocate for finding efficient moves. They apply the mechanisms of neighbor lists and marking for speed improvement. To enhance the solution quality, they combine this procedure with a Guided Local Search heuristic. Reed et al. (2014) use an Ant Colony System to solve CVRP and MCVRP for waste collection from households. Henke et al. (2015) present a model for this problem with flexible compartment sizes and solve the problem by a Variable Neighborhood Search.

Chajakis and Guignard (2003) investigate MCVRP for distribution to convenience stores. They propose mathematical models for two possible cargo space layouts and solve this problem by a Lagrangian Relaxation algorithm. Lahyani et al. (2015) investigate the olive oil collection process in Tunisia as a multi-period MCVRP. They present a mathematical model along with a set of known and new valid inequalities. They use this in a Branch-and-Cut algorithm and evaluate the performance of the algorithm on real data sets under different transportation scenarios.

MCVRP has also been investigated in more general settings: Repoussis et al. (2007) present a hybrid of a Greedy Randomized Adaptive Search procedure and a Variable Neighborhood Search for the heterogeneous fixed fleet No-Split MCVRP. Derigs et al. (2011) present an integer mathematical model for the C-Split MCVRP and propose a range of algorithms that use different approaches for construction, including Local Search, Large Neighborhood Search, Simulated Annealing, Record-to-Record Travel, and Tabu Search. Wang et al. (2014) formalize the heterogeneous fixed fleet multi-compartment vehicle routing problem in the C-Split setting and present a Reactive Guided Tabu Search for solving this problem. Archetti et al. (2014) study different strategies for distributing a set of commodities to customers. They compare the effect of using vehicles dedicated to one type of commodity to using vehicles that carry different commodities in a single compartment. They also consider the option of splitting orders in various ways. They solve small instances with up to 15 customers and 3 compartments to optimality within 30 min by a Branch-and-Cut algorithm. Additional instances with up to 100 customers are solved heuristically by creating multiple copies of each customer based on the commodities requested by the customer, and applying a heuristic for the VRP. Archetti et al. (2015) apply a Branch-and-Price-and-Cut algorithm on the C-Split

MCVRP. They consider two different acceleration heuristics based on a state space relaxation technique and 2-cycle elimination for speeding up the Label Setting Algorithm. The proposed algorithm solves all of the instances with 15 customers, and some instances with up to 40 customers and 3 commodities to optimality within 2 h.

In most of the related papers, heuristics and meta-heuristics are used, but in many cases, neither lower bounds nor optimal solutions exist and therefore, the quality of the obtained solutions is not known. In contrast, Avella et al. (2004), Cornillier et al. (2008a), Lahyani et al. (2015), and Archetti et al. (2015) present exact solution methods for solving the MCVRP. However, Avella et al. (2004) assume that truck compartments are either empty or full and that the number of customers on each route is at most four. Cornillier et al. (2008a) assume that the content of each compartment cannot be split between customers and that the number of customers visited by each truck on any given route will not exceed two. Each of these constraints simplifies the analysis and facilitates applying exact methods. Lahyani et al. (2015) present a Branch-and-Cut algorithm for the multi-period MCVRP, but they have not investigated the problem in the No-Split setting. Archetti et al. (2015) apply a Branch-and-Price-and-Cut algorithm on the C-Split MCVRP. They have not studied this problem in No-Split setting.

3 Mathematical models

In this section, we present mathematical models for the MCVRP. We first consider the C-Split MCVRP and present a model related to this problem in Sect. 3.1. In Sect. 3.2 we then investigate the No-Split MCVRP. The main difference between these models is related to the possibility of multiple visits. In the C-Split delivery, each of the commodities demanded by a customer may be serviced by separate vehicles but in the No-Split delivery, the full set of commodities demanded by a customer should be serviced by a single vehicle.

Consider a complete graph $G(N, E)$ where $N = \{0, 1, \dots, n\}$ is a set of nodes including one depot (node 0) and a set N' of n customers and E is a set of edges. The depot stores a set M of commodities which must be delivered by a homogeneous fleet K of vehicles, each with $|M|$ compartments. Compartment m of each vehicle is dedicated to commodity m and has a known capacity Q_m . Each customer $i \in N'$ has a known demand $q_{im} \leq Q_m$ for each commodity m , possibly null for some commodities not ordered by the customer. We use M_i to denote the set of commodities demanded by customer i , i.e. $M_i = \{m \in M | q_{im} > 0\}$. The travel cost from $i \in N$ to $j \in N$ is given by c_{ij} and is symmetric.

3.1 C-Split MCVRP

In the following, we will focus on the C-Split MCVRP and define

$$y_{imk} = \begin{cases} 1 & \text{if customer } i \in N' \text{ receives commodity } m \in M \text{ from vehicle } k \in K \\ 0 & \text{otherwise.} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{if vehicle } k \in K \text{ travels directly from } i \in N \text{ to } j \in N \\ 0 & \text{otherwise.} \end{cases}$$

The C-Split MCVRP can then be stated as follows:

$$\min \sum_{i \in N} \sum_{j \in N} \sum_{k \in K} c_{ij} x_{ijk} \quad (1)$$

$$\text{st } \sum_{k \in K} y_{imk} = 1 \quad \forall i \in N', \forall m \in M_i \quad (2)$$

$$\sum_{i \in N'} y_{imk} q_{im} \leq Q_m \quad \forall m \in M, \forall k \in K \quad (3)$$

$$y_{imk} \leq \sum_{j \in N} x_{ijk} \quad \forall i \in N', \forall m \in M_i, \forall k \in K \quad (4)$$

$$\sum_{j \in N} x_{0jk} \leq 1 \quad \forall k \in K \quad (5)$$

$$\sum_{j \in N} x_{ijk} = \sum_{j \in N} x_{jik} \quad \forall i \in N', \forall k \in K \quad (6)$$

$$\sum_{i \in N} x_{i0k} \leq 1 \quad \forall k \in K \quad (7)$$

$$\sum_{i,j \in S} x_{ijk} \leq |S| - 1 \quad \forall S \subseteq N', \quad |S| \geq 2, \forall k \in K \quad (8)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i, j \in N, \forall k \in K \quad (9)$$

$$y_{imk} \in \{0, 1\} \quad \forall i \in N', \forall m \in M, \forall k \in K \quad (10)$$

The objective function represents the total cost of the routes to be minimized. Constraints (2) specify that each commodity ordered by a customer is brought by a single vehicle. Constraints (3) ensure that the compartment capacities are respected. Constraints (4) indicate that a vehicle $k \in K$ can only deliver a commodity $m \in M_i$ to customer $i \in N'$ if the vehicle visits that customer. Constraints (5) ensure that each vehicle $k \in K$ leaves the depot at most once. Constraints (6) ensure the continuity of each route. Constraints (7) ensure that the depot is not visited more than once by each vehicle. Constraints (8) are the classical sub-tour elimination constraints and finally, constraints (9) and (10) are the domain constraints.

3.2 No-Split MCVRP

In the No-Split MCVRP, each customer must be serviced by a single vehicle and the vehicle must therefore have enough capacity to service all commodities to be delivered to that customer. The No-Split MCVRP model is proposed as follows:

$$\min \sum_{i \in N} \sum_{j \in N} \sum_{k \in K} c_{ij} x_{ijk} \quad (11)$$

$$\text{st } \sum_{j \in N} \sum_{k \in K} x_{ijk} = 1 \quad \forall i \in N' \quad (12)$$

$$\sum_{i \in N'} q_{im} \sum_{j \in N} x_{ijk} \leq Q_m \quad \forall m \in M, \forall k \in K \quad (13)$$

Constraints (5)–(9)

The objective function represents the total routing cost to be minimized. Constraints (12) ensure that each customer is visited exactly once. Constraints (13) define upper bounds on the delivery amounts determined by the capacities of the vehicle compartments.

3.3 Set partitioning model

Because we solve the two problems by Branch-and-Price, we reformulate them in terms of set partitioning models. We first consider the C-Split MCVRP. A set partitioning model can be obtained by applying Dantzig–Wolfe decomposition to (1)–(10) (Dantzig and Wolfe 1960, 1961). Due to the structure of the problem, a natural choice is to keep constraints (2) in the master problem resulting from the decomposition as these are the only constraints which include all vehicles. This leaves constraints (3)–(10) for the so-called pricing problem of the decomposition.

Formally, for each vehicle $k \in K$, we consider the polytope $\mathcal{P}_k$ given by constraints (3)–(10) with fixed k . Because the vehicles are identical, these polytopes are identical and we can therefore consider a single polytope defined as $\mathcal{P} = \mathcal{P}_k, \forall k \in K$. With this notation, $\mathcal{P}$ defines the feasible region of the pricing problem of the decomposition and we define Ω to be the set of extreme points of $\mathcal{P}$. Such an extreme point corresponds to a route for a single vehicle, that starts and ends in the depot node, visits a number of customers where it delivers some of the requested commodities (possibly all) while respecting the compartment capacities. As Branch-and-Price is based on delayed column generation, the complete set of routes in Ω will not be considered. Rather, routes will be created dynamically in the pricing problem based on their ability to improve the solution.

Let c_r be the cost of such a route $r \in \Omega$ and let a_{imr} be a parameter taking the value 1 if route $r \in \Omega$ delivers commodity $m \in M$ to customer $i \in N'$, and zero otherwise. In order to construct the master problem for the C-Split MCVRP, we define for each route $r \in \Omega$ the decision variable λ_r as 1 if route r is used in the solution, and zero otherwise. The C-Split MCVRP can then be formulated as the

following set partitioning model, where the convexity constraint (16) is relaxed from equality because we do not need to use all vehicles and (15) ensure that all requested commodities are serviced at all customers.

$$\min \sum_{r \in \Omega} c_r \lambda_r \quad (14)$$

$$\text{st } \sum_{r \in \Omega} a_{imr} \lambda_r = 1 \quad \forall i \in N', \forall m \in M_i \quad (15)$$

$$\sum_{r \in \Omega} \lambda_r \leq |K| \quad (16)$$

$$\lambda_r \in \{0, 1\} \quad \forall r \in \Omega \quad (17)$$

For the LP-relaxation of this model, let $\pi_{im}, \forall i \in N', \forall m \in M_i$ be the dual variables corresponding to constraints (15) and let μ be the dual variable of constraint (16). Then, letting y_{im} and x_{ij} be simplifications of y_{imk} and x_{ijk} , respectively, the objective function of the pricing problem becomes

$$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} - \sum_{i \in N'} \sum_{m \in M_i} \pi_{im} y_{im} - \mu \quad (18)$$

For the No-Split MCVRP, we use a similar approach. Here, constraints (12) are the only constraints that include multiple vehicles. Hence, these constraints take the place of (2) in the master problem while (5)–(9) along with (13) are treated in the pricing problem. For the No-Split MCVRP, we remove the commodity-index because all commodities requested by a customer must be serviced by the same vehicle. The parameter a_{ir} therefore takes the value 1 if route $r \in \Omega$ services customer $i \in N'$, and zero otherwise, and the master problem is as follows:

$$\min \sum_{r \in \Omega} c_r \lambda_r \quad (19)$$

$$\text{st } \sum_{r \in \Omega} a_{ir} \lambda_r = 1 \quad \forall i \in N' \quad (20)$$

$$\sum_{r \in \Omega} \lambda_r \leq |K| \quad (21)$$

$$\lambda_r \in \{0, 1\} \quad \forall r \in \Omega \quad (22)$$

Letting $\pi_i, \forall i \in N'$ be the dual variables corresponding to constraints (20) and μ be the dual variable of constraint (21) and using a similar simplified notation for x_{ij} , we get the following objective function for the No-Split MCVRP pricing problem:

$$\min \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij} - \sum_{i \in N'} \sum_{j \in N} \pi_i x_{ij} - \mu \quad (23)$$

4 Branch-and-Price algorithm

This section describes our Branch-and-Price algorithm in detail. The overall process is outlined in Fig. 1. The algorithm is initialized by a feasible solution obtained from a construction heuristic and improved by a simulated annealing. This is included in the *start* part of the algorithm and is described in Sect. 4.1.

The lower left part of the figure shows our node selection strategy which is explained in Sect. 4.2. To control the node selection, the algorithm uses two counters: *it* counts the total number of iterations of the full algorithm, and *cnt* counts the number of iterations since the last application of the *depth-first* search. In Sect. 4.2, we also explain our branching strategy. The remainder of the figure follows a relatively classical Branch-and-Bound structure. In the figure, *best solution* refers to the best feasible integer solution found so far.

Fig. 1 Flow of the overall process

The *pricing* part of the algorithm comes into play when a node in the branching tree is being solved. We explain our overall strategy for solving a node in Sect. 4.3, and Fig. 2 provides an overview of the process, which includes an exact column generation described in Sect. 4.4 and a number of heuristic column generation algorithms that are detailed in Sect. 4.5.

4.1 Initialization

We initialize our Branch-and-Price algorithm by defining an LP model containing a small number of rows and columns. For the C-Split MCVRP, the LP is initialized by $n|M|$ constraints of type (15) to ensure that all commodities are serviced at all customers, in addition to the constraint (16) to limit the fleet of vehicles. For the No-Split MCVRP, the $n|M|$ constraints of type (15) are replaced by n constraints of type (20) to ensure that all customers are serviced.

To ensure that the LP model has a feasible solution, even when we delete some columns due to branching rules, we add a number of dummy columns to the problem. These columns are never deleted, but each dummy column has a large cost to ensure that it will not become part of the final solution. However, a feasible LP solution can always be obtained by setting the variables corresponding to the dummy columns to one. In the C-Split MCVRP, to ensure feasibility for every customer i -commodity m combination, we create a dummy column that represents a route which starts at the depot, travels along a shortest path to i , services commodity m , and returns to the depot. The number of dummy columns for the No-Split MCVRP, decreases to n and each column represents a route that starts at the depot, services customer $i \in N'$, and returns to the depot. The coefficient of constraint (16) and constraint (21), respectively, is 0 for the dummy columns to ensure feasibility throughout the algorithm.

In addition to these dummy columns, we use a heuristic for initialization of the master problem. The heuristic starts by applying a nearest neighbor strategy that

Fig. 2 Flow of node solution process

works as follows. Each route starts at the depot and repeatedly extends from its current point to the nearest unvisited customer subject to the vehicle compartment capacities, where nearest is measured in terms of travel cost. For the No-Split version, we have to deliver all commodities ordered by the customer and consequently we only consider customers whose total orders do not exceed the remaining vehicle capacity. For the C-Split we do not need to service all commodities, and the remaining capacity therefore only needs to be sufficient for the customer's demand for one of the commodities. However, when a customer is visited, the vehicle services the customer as regards all the commodities for which the capacity is sufficient. In case some commodities are not serviced at a customer, that customer will still be regarded as unserved with respect to the non-delivered commodities. The route extension continues until there is no more capacity for serving customers or until all customers have been visited, at which point the vehicle returns to the depot. If there is at least one unvisited customer, a new route is initialized.

Once a feasible solution is obtained by this approach, it is improved by a Simulated Annealing using three neighbourhood structures: reallocate, which moves a customer from one position to another; exchange, which swaps two customers; and 2-opt. All three neighborhoods are used in random versions, i.e. the customer nodes to be involved in the neighborhoods are selected randomly. In each iteration of the algorithm, one of the three neighborhoods is chosen randomly with equal probability and is used in that iteration. The Simulated Annealing is run for approximately 30 s, and the columns corresponding to the routes of the final solution are used in the initialization of the master problem.

4.2 Branching and node selection

In our Branch-and-Price algorithm, we apply the branching strategy known as follow-on branching, originally based on Ryan–Foster branching. The idea is to select a pair of required customer nodes, i and j , and create two branches in the branching tree. In the left branch, the connection from i to j is fixed, and j must be serviced immediately after i . In the right branch, this connection is forbidden, and j may not be serviced immediately after i . See Ryan and Foster (1981) and Vanderbeck (2000) for more on follow-on branching.

However, if the number of vehicles used in the optimal solution of the LP-relaxation of the master problem is significantly fractional, we give priority to branching on the number of vehicles. Let v be the number of vehicles used in the solution. We branch on the number of vehicles if $\lfloor v \rfloor + 0.2 < v$ and $v < \lceil v \rceil - 0.2$, where the threshold of 0.2 is found experimentally to be a reasonable choice. In this case, we tighten constraint (16) to require that the number of vehicles is $\leq \lfloor v \rfloor$ in the left branch and $\geq \lceil v \rceil$ in the right branch.

Due to the limitations of the standard node selection algorithms, i.e. breadth-first or depth-first search, we have used a combination of the two that allows a single search algorithm to have the complementary strengths of both. The algorithm starts in a breadth-first search fashion where the node to be explored next is selected among unexplored nodes as the one whose parent's LP-relaxation has the lowest

value. In case of a tie we select an arbitrary node. This is repeated for b_1 iterations. Next, a depth-first search explores nodes along a left branch until it either reaches a node that has an integer solution or a node that has an LP-relaxation value that is worse than the current best integer solution. At this point the depth-first search stops. Then the algorithm proceeds with breadth-first search and the process is repeated until there are no more nodes for branching or the global lower bound exceeds the value of the best feasible solution. However, after 500 iterations, the algorithm shifts to pure breadth-first search. To optimize the performance of the program we tested different values and chose $b_1 = 18$. The node selection strategy is outlined in the lower left corner of Fig. 1. In this algorithm, the left branch is chosen before the right branch.

4.3 Node solving procedure

As is usual in column generation approaches, we create a copy t of the depot node and search for routes starting in node 0, ending in node t , which respect all resource constraints and have negative reduced cost with respect to the current dual prices of the master problem. The column generation is initialized with a resource constraint for each commodity. During the process, additional resource constraints are added as explained below. We let S be the set of all such feasible routes with negative reduced cost and define $S_2 \subseteq S$ as the set of feasible elementary routes with negative reduced costs. Finally, we set $S_1 = S \setminus S_2$, i.e. the set of non-elementary routes. This means that S_2 includes only the routes that visit each customer at most once, whereas routes in S_1 visit at least one customer more than once. Hence, routes in S_1 are not accepted as part of the final solution, but they may be created by the column generation algorithms nonetheless.

We define $\Psi \subseteq N'$ as a set of nodes which will not be allowed to appear on a route more than once in our exact algorithm. As will be explained in Sect. 4.4, the nodes $i \in \Psi$ will be treated as resources and therefore, we seek to keep $|\Psi|$ small.

Figure 2 shows an outline of our procedure for solving a node in the branching tree. Throughout the process, we only add elementary routes to the master problem. First, we iteratively apply heuristic column generation based on $k_{\max}$ heuristics. These are explained in Sect. 4.5. Next, we apply exact column generation where the nodes in Ψ are treated as resources and nodes are added to Ψ when necessary. The exact algorithm is described in Sect. 4.4. At any time, if columns are added to the master problem, the procedure starts with the first heuristic again.

Given a set S_2 of feasible elementary routes with negative reduced costs created by one of the heuristics or by the exact algorithm, we add columns to the master problem. If $|S_2| \leq 150$, we add all routes from S_2 to the master problem. Otherwise, we add the 150 routes from S_2 with the highest negative reduced cost to the master problem.

4.4 Exact column generation

We use a Label Setting Algorithm to solve the pricing problem. By means of this algorithm we find new columns that can improve the master problem or we prove

that no such column exists. The algorithm identifies the full set of non-dominated columns which are partitioned into S_1 and S_2 . If $S_2 \neq \emptyset$, we add up to 150 of these columns representing elementary routes to the master problem. The columns added are those with highest negative reduced cost. If the algorithm did not identify any elementary routes, i.e. $S_2 = \emptyset$, but found non-elementary routes with negative reduced cost, i.e. $S_1 \neq \emptyset$, the algorithm proceeds by adding additional resource constraints to the problem. This process is explained below.

We refer the reader to Irnich and Desaulniers (2005) for a thorough description of labeling algorithms for solving shortest path problems with resource constraints and to Desaulniers et al. (2005) for a general introduction to column generation.

The algorithm uses a set $\mathcal{L}_i$ of labels for each node $i \in N \cup \{t\}$. Each label L_i^k in $\mathcal{L}_i$ represents a partial route R_i^k from node 0 to node i . Associated with L_i^k is a capacity resource vector σ_i^k , where σ_{im}^k represents the capacity consumption of commodity m along R_i^k and a 'node' resource vector W_i^k , where W_{ij}^k gives the number of visits to node j on R_i^k . Note that $W_{ii}^k \geq 1$ because the label is in node i , and $W_{ij}^k \leq 1 \ \forall j \in \Psi$. Finally, we use $C(R_i^k)$ to denote the reduced cost of the partial route R_i^k . These are determined by applying (18) and (23) to the partial route R_i^k for C-Split and No-Split, respectively.

A label L_i^a dominates a label L_i^b if any feasible extension into a complete route that can be made from L_i^b can also be made from L_i^a , and if routes resulting from extensions of L_i^b cannot have smaller reduced costs than the routes obtained from the same extensions of L_i^a . In this case, the label L_i^b is superfluous and can be dominated. Formally, a label L_i^a dominates a label L_i^b if and only if $\sigma_i^a \leq \sigma_i^b$, $C(R_i^a) \leq C(R_i^b)$, and $W_{ij}^a \leq W_{ij}^b$, $\forall j \in \Psi$. To explain the latter, assume that $\Psi = \{1, 2, 4, 6, 8\}$ and label L_i^a services route $R_i^a = [1, 2, 3, 8]$. Consider label L_i^b and assume that $\sigma_i^a \leq \sigma_i^b$, $C(R_i^a) \leq C(R_i^b)$. If the route of L_i^b is $R_i^b = [1, 2, 6, 8]$, then L_i^a can dominate L_i^b because $R_i^a \cap \Psi = [1, 2, 8] \subseteq [1, 2, 6, 8] = R_i^b$. Here, node 3 is not relevant because it is not in Ψ . But if the route of L_i^b is $R_i^b = [1, 2, 4, 8]$, then L_i^a cannot dominate L_i^b because $W_{i6}^a = 1$ but $W_{i6}^b = 0$.

To handle the branching rules described in Sect. 4.2, we define an $n(n+1)$. *tabu matrix* T and a *fix vector* F of size $n+1$.

We first consider the right-branch rule, forbidding the service of a required node j or a return to the depot to follow immediately after another required node i . This is recorded in T as follows:

$$T_{ij} = \begin{cases} 1 & \text{if } j \in N' \cup \{t\} \text{ may not be visited immediately after } i \in N \\ 0 & \text{otherwise.} \end{cases}$$

In the left branch, we set $F_i = j$, thereby forcing j to follow immediately after i . In the labeling algorithm, before extending L_i^k to j , T_{ij} and F_i are checked and appropriate action is taken.

Algorithm 1 Label Setting Algorithm

```

Create  $L_0^1$  with  $R_0^1 = 0, \sigma_0^1 = 0, W_0^1 = 0$ , and  $C(R_0^1) = 0$ .
Set  $\mathcal{L}_0 = \{L_0^1\}$ 
Set  $\mathcal{L}_i = \emptyset, \forall i \in N' \cup \{t\}$ .
while  $\bigcup_{i \in N} \mathcal{L}_i \neq \emptyset$  do
    Choose a label  $L_i^k \in \bigcup_{i \in N} \mathcal{L}_i$  and set  $\mathcal{L}_i = \mathcal{L}_i \setminus \{L_i^k\}$ 
    for all  $j \in N' \cup \{t\}$  do
        if  $T_{ij} \neq 1$  and  $(F_i = j \text{ or } F_i = 0)$  then
            Let  $L_j'$  denote the label obtained by extending  $L_i^k$  to  $j$ .
            if  $L_j'$  is feasible with respect to resources then
                if  $L_j'$  is not dominated by any label in  $U_j$  then
                    Create  $L_j'$  and set  $\mathcal{L}_j = \mathcal{L}_j \cup \{L_j'\}$ .
                    Remove any dominated labels from  $\mathcal{L}_j$ 
                end if
            end if
        end if
    end for
end while
Set  $S_1 = \{L_i^k \in \mathcal{L}_i \mid \exists j \in N' \text{ with } W_{ij}^k > 1 \text{ and } C(R_i^k) < 0\}$ 
Set  $S_2 = \{L_i^k \in \mathcal{L}_i \mid W_{it}^k \leq 1 \forall j \in N' \text{ and } C(R_i^k) < 0\}$ 

```

Algorithm 1 shows the details of the proposed Label Setting Algorithm. The algorithm starts with an empty set of labels, except for a zero-label for the depot. It then repeatedly selects an unprocessed label L_i^k with smallest resource consumption with respect to commodity 1, σ_{i1}^k . For all $j \in N'$ a new label L_j' is created as an extension of L_i^k to j if it is resource feasible with respect to the capacity and 'node' resources, and allowed by the branching rules. Similarly, all labels are extended to node t , if feasible, representing a return to the depot. This continues until there are no unprocessed labels.

Algorithm 2 Exact Column Generation

```

 $it = 1$ ,
while  $it = 1$  or  $S \neq \emptyset$  do
    Run the Label Setting Algorithm (Algorithm 1),
    if  $S_2 \neq \emptyset$  then
        return  $S_2$ 
    else if  $S_1 \neq \emptyset$  then
        Set  $L_t^{k*} = \operatorname{argmin}_{L_t^k \in S_1} \{C(R_t^k)\}$ .
        Set  $j^* = \operatorname{argmax}_{j \in N'} \{W_{tj}^{k*}\}$ 
         $\Psi = \Psi \cup \{j^*\}$ .
    end if
     $it = it + 1$ 
end while
return  $\emptyset$ 

```

To strengthen the formulation, we only consider elementary routes in the master problem. Since the problem of finding elementary routes is strongly NP-hard, we use the idea of Boland et al. (2006) and employ the Label Setting Algorithm for solving the Resource Constrained Shortest Path Problem with node resources for only some nodes. We set $\Psi = \emptyset$ and solve the non-elementary problem using the Label Setting Algorithm. If no elementary route is created, we examine the non-elementary route with the highest negative reduced cost, R_i^k . In this route, we identify a customer j with $W_{ij}^k > 1$, i.e. customer j is visited more than once. Based on this, we set $\Psi = \Psi \cup \{j\}$. We repeat the procedure until there are no more non-elementary routes ($S_1 = \emptyset$), at which point the node in the branching tree has been solved to optimality. This exact algorithm is outlined in Algorithm 2.

4.5 Heuristic column generation

The main weakness of the exact algorithm is that the running time increases as the number of nodes treated as resources grows, because it becomes increasingly more difficult to dominate labels. This is the motivation for only running the exact column generation to ensure that there are no more elementary routes with negative reduced cost and hence prove that the node has been solved to optimality. For the No-Split problem, we use $k_{\max} = 4$ different heuristic techniques in order to speed up the Branch-and-Price algorithm, whereas we use $k_{\max} = 5$ for the C-Split version of the problem. Each of these heuristics produces elementary routes, but not necessarily the routes with the most negative reduced cost.

The first heuristic is a simple heuristic column generation based on the nearest neighbor heuristic which works as follows. We start a route at the depot and iteratively extend the route from its current end point to the nearest unvisited customer subject to the vehicle compartment capacity in the same way as we did in the construction heuristic described in Sect. 4.1. However, for this heuristic column generation, 'nearest' is measured in terms of travel cost minus the cost π_{im} for each delivery of commodity m to customer i (π_i for No-Split). The route is extended as long as the reduced cost is negative and the remaining capacity of the vehicle is large enough to service at least one of the unvisited customers.

The second heuristic is based on scaling of the capacity of the compartments. The exact algorithm is executed with modified vehicle capacities $Q'_m = g_1 Q_m$, ($\forall m \in M, 0 < g_1 \leq 1$). In effect, this means that a vehicle can service fewer customers. Even though such shorter routes do not fully use the vehicle capacity, they can still be useful as supplements to longer routes. This heuristic is efficient for instances with high capacity of the vehicle compartments compared to the demand of customers because these are the instances where the labeling algorithm is generally time consuming. After testing different values, $g_1 = 0.5$ was selected.

The third heuristic is also a truncated version of our labeling algorithm and is based on extending each label from each node to its neighbors within a predefined distance. Each label in node i is only extended to the nodes j to which the distance c_{ij} is below a specified length D . The length of D is calculated by

$\max_{i,j \in N} \{c_{ij} g_2\}$, $0 < g_2 \leq 1$. By considering only edges within this maximum length D , the algorithm will be limited to a restricted set of neighbors when processing each node. If the algorithm fails to identify negative reduced cost routes, g_2 is increased to αg_2 and the search is repeated while $g_2 \leq 1$. After computational testing, the initial values $g_2 = 0.6$ and $\alpha = 1.3$, were chosen.

The fourth heuristic is a heuristic Label Setting Algorithm with modified dominance rules. In this algorithm, a label L_i^a can dominate label L_i^b if it dominates with respect to the remaining capacity of compartments and the cost of the route, independent of 'node' resource vector. In other words, L_i^a can dominate L_i^b if and only if $\sigma_i^a \leq \sigma_i^b$, and $C_i^a \leq C_i^b$. By applying this heuristic, the valid routes are created with better speed.

The fifth heuristic is only used for the C-Split problem. In this heuristic, the labeling algorithm is executed for the C-Split problem but with No-Split settings. This means that we only extend a label L_i^k from i to customer j , if the remaining capacity is sufficient to service all commodities at customer j , i.e. if $\sigma_{im}^k + q_{jm} \leq Q_m \forall m \in M_j$. This heuristic creates valid routes similar to routes in the No-Split setting and helps us to reduce the running time because the commodities are not treated separately.

5 Computational experiments

We have tested our algorithm on a total of 274 instances. In this section, we present our computational results and analysis, which form the basis of our conclusions. The algorithm is implemented in C++ in Microsoft Visual Studio 2010 with the use of IBM ILOG CPLEX 12.6.1 callable library. All tests were made on a laptop with a Pentium core i5 processor and a clock speed of 1.8 GHz and 8 GB of RAM.

Details about our test sets are presented in Sect. 5.1. We have generated five sets of test instances which differ in type of demand and vehicle capacity. They are indexed 1 through 5. Furthermore, we have used three sets of instances from Muyldermans and Pang (2010). These are referred to as M4, M5, and M6, referring to the authors' own numbering.

Our results are structured as follows. In Sect. 5.2, we analyze the performance of our algorithm as regards its ability to solve problems using the No-Split strategy, and Sect. 5.3 is devoted to the performance as regards the C-Split strategy. From a computational perspective, the No-Split strategy is the easier of the two strategies for a Branch-and-Price algorithm due to the reduced flexibility for service. Therefore, we only present results for the C-Split strategy up to the point where the instances become too large for the algorithm to handle. For the C-Split, the size is perceived as a combination of nodes and commodities. This is further explained in Sect. 5.3. Finally, in Sect. 5.4, we compare the two strategies and analyze if allowing C-Split can lead to reductions in cost.

Table 1 provides a summary of the main results and an overview of Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15. For each of the tables, Table 1 provide the name of the test set along with the type of demand in the instances and the vehicle

Table 1 Summary of computational results

Tables	Set	Demand	Capacity	No	Opt	Unsolved	Av gap	Max gap	Strategy
2	1	u(1, 5)	10	44	28	0	0.5	3.1	No-Split
3	2	u(1, 5)	15	24	1	4	3.5	16.6	No-Split
4	3	u(1, 5)	20	24	1	12	6.1	16.5	No-Split
5	M5	u(1, 5), u(6,10)	6–12, 12–36	35	16	0	1.6	7.0	No-Split
6	M6	u(1, 5), u(6,10)	6–12, 12–36	35	13	0	1.2	6.4	No-Split
7+8	4	bin	3	73	48	0	0.7	4.8	No-Split
9	5	bin	4	24	1	8	3.7	7.2	No-Split
10	M4	bin	3	15	0	0	7.9	24.5	No-Split
11	1	u(1, 5)	10	20	13	0	0.6	3.5	C-Split
12	M5	u(1, 5), u(6,10)	6, 12–18	15	0	0	11.1	29.2	C-Split
13	M6	u(1, 5), u(6,10)	6, 12–18	15	1	0	5.6	13.4	C-Split
14+15	4	bin	3	55	22	0	1.6	6.5	C-Split

capacity for each commodity. The center part of the table states the number of instances in the set (NO.), the number of instances solved to optimality within a time limit of 1 h (Opt.), the number of instances with unsolved root after the time limit of 1 h (Unsolved), and the average and maximal gaps after 1 h of computation. Finally, the right-most column gives information about the strategy used in the tests presented in the tables.

Tables 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, and 15 give the details of our results. In these tables, we report the following: the instance name, which is composed of four numbers: the number of customers in the graph, the number of commodities, the vehicle capacity for each commodity, and finally an ID number to distinguish instances with similar characteristics. Under the *Root* header, we report the initial feasible solution obtained by running a simulated annealing algorithm for approximately 30 s (*SA*), the lower bound obtained as the LP-relaxation of the root of the branching tree (*LB*), and the percentage gap between this lower bound and the final best solution obtained (*Gap*). Under the *Best known* header, we report the value of the best feasible solution found (*UB*), the global lower bound (*LB*), and the percentage gap between the two (*Gap*). The gap is replaced by an asterisks (*) if optimality has been proven. Finally, we report the number of nodes in the branching tree (*Nodes*), the maximum number of customers on a route in the best solution (*Cust*), the number of vehicles used in the best solution (*Veh*), and the total running time in seconds (*Time*). If the time limit of 1 h is reached, we allow the algorithm to finish solving the current node in the branching tree and use TL to denote the time.

5.1 Data instances

We have generated a total of 189 new data instances which have been grouped into 5 sets. In addition, we have performed tests on 85 instances from Muylldermans and Pang (2010), grouped into 5 sets. We provide the details of the data in the following.

Table 2 No-Split strategy for U(1, 5) demand and a capacity of 10

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-10-1	503.3	495.8	1.5	503.3	503.3	*	3	3	4	1
10-2-10-2	432.2	410.2	*	410.2	410.2	*	0	4	3	11
10-2-10-3	461.2	461.2	*	461.2	461.2	*	0	3	4	1
10-2-10-4	451.7	436.8	3.4	451.7	451.7	*	7	4	4	2
10-2-10-5	420.1	396.1	*	396.1	396.1	*	0	4	3	1
15-2-10-1	641.7	626.6	0.6	630.2	630.2	*	3	4	5	13
15-2-10-2	594.4	573.8	1.1	588.6	588.6	*	23	4	5	15
15-2-10-3	553.1	479.6	5.5	501.3	501.3	*	84	3	5	17
15-2-10-4	656.9	650.5	1.0	656.9	656.9	*	7	3	7	2
15-2-10-5	572.3	525.3	*	525.3	525.3	*	0	3	6	2
20-2-10-1	878.3	856.2	0.2	857.6	857.6	*	3	3	8	2
20-2-10-2	636.2	623.4	0.8	628.5	628.5	*	13	4	6	4
20-2-10-3	666.1	652.5	2.1	666.1	666.1	*	73	4	7	14
20-2-10-4	900.8	883.1	1.8	899.4	899.4	*	9	3	9	3
20-2-10-5	684.1	674.7	1.4	684.1	684.1	*	5	4	8	2
25-2-10-1	1022.8	992.6	3.0	1008.5	1008.5	*	859	4	10	113
25-2-10-2	768.8	743.4	2.0	753.8	753.8	*	101	4	7	25
25-2-10-3	882.2	848.2	2.0	855.4	855.4	*	68	5	10	17
25-2-10-4	1073.0	1045.3	0.9	1055.2	1055.2	*	23	3	11	5
25-2-10-5	852.3	839.5	1.5	852.3	852.3	*	17	4	9	6
40-2-10-1	1510.7	1416.3	1.8	1441.9	1441.9	*	1111	4	13	1526
40-2-10-2	1352.4	1233.6	2.0	1258.3	1258.3	*	1966	4	13	1616
40-3-10-1	1559.4	1489.5	0.6	1497.9	1497.9	*	183	4	15	579
40-3-10-2	1315.0	1234.2	1.4	1251.4	1251.4	*	783	4	15	1571
40-4-10-1	3383.1	2913.6	0.1	2917.4	2917.4	*	55	4	12	307
40-4-10-2	3328.7	3063.0	0.2	3069.2	3069.2	*	75	4	13	36
50-2-10-1	1898.1	1771.4	1.4	1795.5	1783.6	0.7	4219	5	16	TL
50-2-10-2	1571.8	1438.5	2.2	1470.7	1451.8	1.3	940	4	16	TL
50-3-10-1	1842.9	1701.5	3.1	1755.1	1716.4	2.3	7590	4	17	TL
50-3-10-2	1700.1	1576.9	0.5	1584.9	1584.9	*	525	4	18	490
50-4-10-1	4196.0	3607.2	2.2	3686.0	3615.6	1.9	1571	4	16	TL
50-4-10-2	4172.3	3663.4	0.5	3683.2	3683.2	*	888	4	16	976
75-2-10-1	2639.7	2447.3	2.6	2511.3	2455.3	2.3	1935	4	25	TL
75-2-10-2	2528.6	2269.1	2.1	2317.0	2276.2	1.8	629	5	25	TL
75-3-10-1	2830.6	2489.2	1.6	2528.5	2500.5	1.1	2976	4	26	TL
75-3-10-2	2729.9	2409.2	0.7	2426.2	2418.4	0.3	2660	4	26	TL
75-4-10-1	5915.2	5356.2	0.6	5389.0	5357.6	0.6	754	4	23	TL
75-4-10-2	6144.7	5252.5	0.1	5259.2	5256.5	0.1	1338	4	23	TL
100-2-10-1	3413.1	2996.8	1.6	3043.8	3001.6	1.4	620	4	31	TL
100-2-10-2	3230.1	2812.8	2.2	2874.4	2821.1	1.9	718	5	32	TL

Table 2 continued

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-3-10-1	3588.1	3187.5	1.0	3219.5	3190.7	0.9	475	4	34	TL
100-3-10-2	3531.8	3101.9	3.1	3199.5	3102.2	3.1	273	4	35	TL
100-4-10-1	8040.5	7122.0	0.4	7150.8	7122.2	0.4	1221	4	31	TL
100-4-10-2	8206.2	7255.7	0.5	7295.3	7263.8	0.4	276	4	32	TL

Table 3 No-Split strategy for U(1, 5) demand and a capacity of 15

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-15-1	1064.8	945.3	3.9	982.6	955.2	2.9	719	7	8	TL
40-2-15-2	1073.0	1004.6	4.1	1045.4	1020.0	2.5	1126	6	9	TL
40-3-15-1	1208.2	1008.0	4.8	1056.5	1022.5	3.3	860	6	10	TL
40-3-15-2	1111.7	999.7	5.0	1050.2	1014.6	3.5	871	6	9	TL
40-4-15-1	2511.9	2246.8	3.0	2317.2	2252.1	2.9	677	5	10	TL
40-4-15-2	2304.1	2077.8	1.4	2107.9	2107.9	*	173	5	9	193
50-2-15-1	1268.9	1167.0	2.0	1189.9	1171.4	1.6	329	7	11	TL
50-2-15-2	1276.7	1176.9	3.3	1215.9	1180.6	3.0	245	6	11	TL
50-3-15-1	1370.6	1220.2	3.1	1258.2	1226.0	2.6	230	5	12	TL
50-3-15-2	1464.4	1324.3	1.1	1338.3	1329.5	0.7	289	5	13	TL
50-4-15-1	3082.4	2582.9	1.4	2620.7	2588.5	1.2	244	5	11	TL
50-4-15-2	2714.7	2484.3	2.1	2536.1	2511.2	1.0	114	6	11	TL
75-2-15-1	1902.7	1647.4	4.5	1722.3	1653.1	4.2	766	6	16	TL
75-2-15-2	1913.4	1704.7	5.2	1794.0	1705.3	5.2	138	6	17	TL
75-3-15-1	1896.5	1703.0	4.1	1772.0	1704.4	4.0	330	6	17	TL
75-3-15-2	1997.9	1761.1	2.6	1807.7	1763.4	2.5	201	6	18	TL
75-4-15-1	4067.5	—	—	4067.5	—	—	0	5	16	TL
75-4-15-2	4197.1	—	—	4197.1	—	—	0	5	16	TL
100-2-15-1	2604.7	2134.3	6.3	2267.7	2134.4	6.2	306	7	23	TL
100-2-15-2	2404.4	2061.6	16.6	2404.4	2061.6	16.6	55	7	22	TL
100-3-15-1	2530.4	2167.2	4.8	2272.1	2172.2	4.6	382	5	23	TL
100-3-15-2	2728.9	2214.3	1.6	2250.3	2214.5	1.6	218	5	23	TL
100-4-15-1	5447.2	—	—	5447.2	—	—	0	6	22	TL
100-4-15-2	5405.0	—	—	5405.0	—	—	0	6	22	TL

Table 4 No-Split strategy for U(1, 5) demand and a capacity of 20

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-20-1	907.8	775.6	6.8	828.7	779.4	6.3	354	7	7	TL
40-2-20-2	997.4	781.9	0.4	785.5	785.5	*	81	6	7	3030
40-3-20-1	922.8	782.7	8.7	850.8	785.8	8.3	488	7	7	TL
40-3-20-2	977.9	838.7	2.3	858.2	841.1	2.0	461	7	7	TL
40-4-20-1	1748.7	—	—	1748.7	—	—	0	6	7	TL
40-4-20-2	1857.0	—	—	1857.0	—	—	0	7	7	TL
50-2-20-1	977.5	895.9	9.1	977.5	895.9	9.1	67	8	8	TL
50-2-20-2	995.7	942.5	4.0	980.1	947.3	3.5	331	7	9	TL
50-3-20-1	1170.4	1009.1	2.0	1029.1	1009.1	2.0	230	7	9	TL
50-3-20-2	1055.0	954.1	3.9	991.7	954.1	3.9	197	6	10	TL
50-4-20-1	2186.9	—	—	2186.9	—	—	0	7	8	TL
50-4-20-2	2099.9	—	—	2099.9	—	—	0	7	8	TL
75-2-20-1	1616.7	1348.4	6.8	1440.3	1348.4	6.8	122	8	12	TL
75-2-20-2	1551.2	1331.5	16.5	1551.2	1331.5	16.5	51	7	14	TL
75-3-20-1	1619.9	1446.2	9.7	1586.9	1446.2	9.7	264	7	13	TL
75-3-20-2	1507.0	1324.9	13.7	1507.0	1326.0	13.6	64	7	13	TL
75-4-20-1	3323.9	—	—	3323.9	—	—	0	7	13	TL
75-4-20-2	3204.2	—	—	3204.2	—	—	0	7	12	TL
100-2-20-1	2116.0	—	—	2116.0	—	—	0	8	16	TL
100-2-20-2	1809.5	—	—	1809.5	—	—	0	8	16	TL
100-3-20-1	2036.0	—	—	2036.0	—	—	0	8	17	TL
100-3-20-2	1912.0	—	—	1912.0	—	—	0	8	17	TL
100-4-20-1	4288.7	—	—	4288.7	—	—	0	7	18	TL
100-4-20-2	4110.2	—	—	4110.2	—	—	0	7	18	TL

We consider the 5 new sets of data first. All customer locations are based on the data of Muyldermans and Pang (2010) and are scattered in a square of size 100×100 distance units where the depot is located in the middle of the square. Muyldermans and Pang (2010) use 100 customers in all instances, but we vary the number of customers in the new sets from 10 to 100. The number of commodities in each instance varies between 2 and 4.

For data sets 1, 2, and 3, the customer's demand for each commodity is drawn uniformly between 1 and 5. This means that each customer has a positive demand for every commodity, i.e. $M_i = M \forall i \in N'$. The vehicle capacity is 10, 15, and 20 for sets 1, 2, and 3, respectively. Therefore, with n fixed, instances in set 1 are expected to be more tractable than those in set 2, and instances in set 3 are, in general, the hardest. Set 1 contains instances down to 10 customers of ensure some relatively easy test instances in the pool, for comparison purposes. Sets 2 and 3 start with 40 customers.

Table 5 No-Split strategy for instances of Fig. 5 of Muyldermans and Pang (2010)

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	5062.3	4891.5	0.1	4897.0	4897.0	*	7	3	59	24
100-2-6-2-A	5021.8	4878.9	0.2	4888.8	4888.8	*	25	2	59	28
100-2-6-3-A	4963.6	4868.8	0.2	4878.6	4878.6	*	99	3	61	57
100-2-6-4-A	5130.1	4927.1	*	4927.1	4927.1	*	0	3	59	27
100-2-6-5-A	5050.4	4766.6	0.3	4778.9	4778.9	*	85	3	55	94
100-2-12-1-A	2785.4	2516.4	4.2	2623.3	2518.5	4.2	263	5	26	TL
100-2-12-2-A	2890.5	2489.1	4.2	2592.6	2490.7	4.1	174	5	28	TL
100-2-12-3-A	2835.3	2470.4	4.1	2570.7	2472.0	4.0	445	6	27	TL
100-2-12-4-A	2900.8	2549.5	4.7	2668.9	2550.0	4.7	148	5	28	TL
100-2-12-5-A	2815.8	2495.9	5.0	2620.3	2496.4	5.0	1170	5	27	TL
100-2-12-1-B	7673.5	7673.5	*	7673.5	7673.5	*	0	2	98	2
100-2-12-2-B	7194.6	7194.6	*	7194.6	7194.6	*	0	2	97	2
100-2-12-3-B	7172.0	7172.0	*	7172.0	7172.0	*	0	2	97	2
100-2-12-4-B	7665.4	7657.8	*	7657.8	7657.8	*	0	2	98	2
100-2-12-5-B	7531.5	7521.2	0.1	7531.5	7531.5	*	7	2	99	2
100-2-18-1-B	4458.0	4309.0	0.6	4334.5	4334.5	*	167	3	50	110
100-2-18-2-B	4145.3	4001.9	0.3	4015.5	4015.5	*	95	2	50	58
100-2-18-3-B	4295.9	4085.5	0.2	4091.7	4091.7	*	67	2	50	100
100-2-18-4-B	4575.6	4252.7	0.1	4256.1	4256.1	*	27	2	50	95
100-2-18-5-B	4349.3	4198.4	0.2	4207.1	4207.1	*	321	2	50	403
100-2-24-1-B	3858.8	3486.2	0.6	3506.2	3496.6	0.3	2272	3	38	TL
100-2-24-2-B	3547.8	3177.0	1.8	3235.7	3179.9	1.8	1218	3	36	TL
100-2-24-3-B	3684.2	3223.6	0.1	3226.7	3226.7	*	119	3	38	565
100-2-24-4-B	3654.9	3277.1	1.6	3330.6	3283.0	1.4	454	3	36	TL
100-2-24-5-B	3779.9	3460.0	1.3	3505.6	3462.8	1.2	440	3	39	TL
100-2-30-1-B	3153.4	2844.6	1.8	2894.4	2844.6	1.8	75	4	30	TL
100-2-30-2-B	2915.5	2622.4	1.9	2673.5	2624.6	1.9	507	4	30	TL
100-2-30-3-B	2998.9	2667.0	3.6	2763.7	2667.1	3.6	67	4	30	TL
100-2-30-4-B	2998.3	2761.9	2.6	2833.2	2763.6	2.5	459	4	30	TL
100-2-30-5-B	3058.8	2818.6	1.6	2863.5	2820.5	1.5	975	4	31	TL
100-2-36-1-B	2616.0	2376.6	2.6	2437.8	2377.0	2.6	98	5	24	TL
100-2-36-2-B	2371.6	2215.8	7.0	2370.2	2215.8	7.0	24	5	26	TL
100-2-36-3-B	2389.3	2250.5	1.0	2273.3	2250.5	1.0	234	5	25	TL
100-2-36-4-B	2546.1	2353.1	1.7	2392.5	2353.1	1.7	254	5	24	TL
100-2-36-5-B	2592.2	2328.9	4.1	2423.6	2328.9	4.1	114	5	25	TL

In data sets 4 and 5, the customer's demand for each commodity is binary and follows the following patterns: If there are 2 commodities in the system, the probability that a customer orders both commodities is 50%. The remaining

Table 6 No-Split strategy for instances of Fig. 6 of Muyldermans and Pang (2010)

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	12,297.2	11,260.4	0.1	11,273.0	11,273.0	*	13	3	50	30
100-2-6-2-A	12,963.1	12,528.9	0.4	12,580.1	12,580.1	*	3	2	58	28
100-2-6-3-A	12,351.3	11,708.2	0.1	11,718.6	11,718.6	*	3	3	52	40
100-2-6-4-A	12,693.9	11,695.0	0.6	11,760.6	11,760.6	*	9	3	54	60
100-2-6-5-A	13,701.0	13,254.7	0.5	13,318.8	13,318.8	*	3	2	62	40
100-2-12-1-A	6119.2	5509.6	3.6	5709.3	5509.6	3.6	249	5	25	TL
100-2-12-2-A	6353.2	5842.0	2.9	6011.3	5843.2	2.9	209	5	27	TL
100-2-12-3-A	6314.1	5599.4	2.5	5740.8	5599.8	2.5	90	6	25	TL
100-2-12-4-A	6063.2	5490.4	2.1	5604.8	5491.2	2.1	196	5	25	TL
100-2-12-5-A	6654.2	6114.4	2.6	6270.4	6115.9	2.5	85	5	28	TL
100-2-12-1-B	21,014.1	21,014.1	*	21,014.1	21,014.1	*	0	2	98	2
100-2-12-2-B	20,784.2	20,784.2	*	20,784.2	20,784.2	*	0	2	97	2
100-2-12-3-B	20,840.2	20,840.2	*	20,840.2	20,840.2	*	0	2	97	2
100-2-12-4-B	21,151.8	21,151.8	*	21,151.8	21,151.8	*	0	1	100	1
100-2-12-5-B	20,647.9	20,647.9	*	20,647.9	20,647.9	*	0	2	98	1
100-2-18-1-B	11,146.9	10,879.1	0.7	10,951.4	10,889.9	0.6	865	2	50	TL
100-2-18-2-B	10,887.7	10,708.8	0.1	10,716.1	10,716.1	*	127	3	49	113
100-2-18-3-B	10,947.6	10,779.0	0.1	10,784.9	10,784.9	*	347	3	49	344
100-2-18-4-B	11,206.4	10,814.8	*	10,818.7	10,818.7	*	97	2	50	169
100-2-18-5-B	11,010.2	10,638.2	0.8	10,723.3	10,654.1	0.6	21	3	50	TL
100-2-24-1-B	8837.6	8092.7	0.1	8099.7	8097.7	0.0	1393	3	36	TL
100-2-24-2-B	8258.2	7593.3	0.4	7626.4	7600.4	0.3	921	3	34	TL
100-2-24-3-B	8657.6	7696.1	0.2	7714.4	7698.6	0.2	390	3	34	TL
100-2-24-4-B	9131.2	8154.9	0.2	8174.0	8159.3	0.2	400	3	37	TL
100-2-24-5-B	8790.0	7717.1	0.8	7776.1	7720.5	0.7	275	3	35	TL
100-2-30-1-B	7165.1	6482.7	1.0	6550.5	6483.3	1.0	198	4	29	TL
100-2-30-2-B	6901.5	6170.6	1.5	6264.1	6171.3	1.5	102	4	28	TL
100-2-30-3-B	6936.3	6274.0	1.0	6337.4	6275.0	1.0	111	4	28	TL
100-2-30-4-B	7024.8	6505.6	2.0	6632.6	6506.8	1.9	284	4	30	TL
100-2-30-5-B	7022.2	6289.5	1.4	6376.8	6289.5	1.4	49	4	28	TL
100-2-36-1-B	5626.2	5287.3	6.4	5626.2	5287.3	6.4	17	5	25	TL
100-2-36-2-B	5627.5	5093.1	1.7	5180.0	5093.1	1.7	104	5	23	TL
100-2-36-3-B	5641.2	5164.1	1.9	5263.2	5164.1	1.9	10	5	23	TL
100-2-36-4-B	5585.9	5300.3	5.4	5585.9	5300.3	5.4	80	5	25	TL
100-2-36-5-B	5547.6	5174.0	3.9	5376.6	5174.0	3.9	296	5	24	TL

customers only order one commodity with each one being evenly represented. If there are 3 commodities in the system, the probability of ordering 1, 2, and 3 commodities is 33%, respectively. Again, once the number has been determined, the

Table 7 No-Split strategy for binary demand and a capacity of 3

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-3-1	411.5	398.9	3.0	410.9	410.9	*	7	4	3	2
10-2-3-2	431.6	424.2	1.7	431.6	431.6	*	3	3	4	1
10-2-3-3	357.0	350.2	1.9	357.0	357.0	*	3	4	3	1
10-2-3-4	421.4	393.2	7.2	421.4	421.4	*	29	4	3	6
10-2-3-5	386.9	386.9	*	386.9	386.9	*	0	4	3	1
10-3-3-1	389.8	375.8	3.7	389.8	389.8	*	5	5	2	2
10-3-3-2	398.3	372.3	1.0	376.0	376.0	*	3	5	3	1
10-3-3-3	421.4	393.2	7.2	421.4	421.4	*	19	4	3	3
10-3-3-4	401.2	393.7	1.2	398.4	398.4	*	15	4	3	3
10-3-3-5	396.5	396.5	*	396.5	396.5	*	0	5	3	1
15-2-3-1	531.8	507.3	3.3	523.8	523.8	*	137	5	4	24
15-2-3-2	550.0	508.5	2.1	518.9	518.9	*	3	4	5	2
15-2-3-3	524.6	507.0	3.4	524.1	524.1	*	37	5	4	8
15-2-3-4	453.2	447.7	1.2	453.2	453.2	*	3	5	4	2
15-2-3-5	554.3	528.4	4.3	550.9	550.9	*	43	4	4	11
15-3-3-1	466.3	460.8	1.2	466.3	466.3	*	3	5	4	2
15-3-3-2	498.1	482.9	1.0	487.5	487.5	*	13	5	4	4
15-3-3-3	518.5	482.8	5.1	507.4	507.4	*	619	5	4	123
15-3-3-4	451.0	451.0	*	451.0	451.0	*	0	5	4	1
15-3-3-5	404.3	380.4	3.5	393.6	393.6	*	85	7	4	19
20-2-3-1	665.2	650.8	0.2	651.9	651.9	*	9	5	5	5
20-2-3-2	613.9	588.6	1.7	598.6	598.6	*	3	4	6	2
20-2-3-3	709.9	704.5	0.8	709.9	709.9	*	17	4	6	5
20-2-3-4	604.7	584.4	2.0	595.6	595.6	*	57	5	6	13
20-2-3-5	568.3	557.3	1.5	565.6	565.6	*	21	5	6	6
20-3-3-1	640.8	599.6	2.1	612.2	612.2	*	275	5	5	65
20-3-3-2	564.1	551.5	2.3	564.1	564.1	*	17	5	5	5
20-3-3-3	691.6	617.6	2.7	634.1	634.1	*	17	4	6	7
20-3-3-4	573.0	562.0	1.9	572.4	572.4	*	29	5	5	23
20-3-3-5	602.1	587.5	1.5	596.1	596.1	*	47	6	5	24
30-2-3-1	867.0	845.6	*	845.6	845.6	*	0	5	9	7
30-2-3-2	892.9	837.4	2.1	853.8	853.8	*	122	4	9	46
30-2-3-3	1027.4	1006.3	1.5	1021.4	1021.4	*	3024	4	9	1615
30-2-3-4	857.5	837.4	2.0	853.8	853.8	*	321	4	9	111
30-2-3-5	826.0	794.1	1.3	804.6	804.6	*	135	5	8	45
30-3-3-1	945.6	882.5	2.9	907.8	907.8	*	4353	5	8	2702
30-3-3-2	859.0	792.2	2.0	808.4	808.4	*	397	5	8	398
30-3-3-3	992.8	947.0	1.6	962.4	962.4	*	91	5	9	66
30-3-3-4	810.2	797.5	1.0	805.1	805.1	*	167	6	8	73

Table 7 continued

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
30-3-3-5	918.2	870.5	2.3	890.3	890.3	*	1378	5	8	2905
40-2-3-1	1197.1	1137.7	*	1138.0	1138.0	*	3	5	10	8
40-2-3-2	1278.2	1074.0	0.6	1079.9	1079.9	*	743	6	11	392
40-2-3-3	1212.0	1144.7	1.8	1165.0	1164.6	0.0	1675	5	11	TL
40-2-3-4	1221.3	1120.0	1.2	1134.0	1134.0	*	55	4	12	220
40-2-3-5	1128.5	1062.2	1.0	1073.0	1073.0	*	306	5	11	326

demand is distributed evenly among the different commodity combinations. For 4 commodities, the probability of ordering 1, 2, 3, and 4 commodities, respectively, is 25%. The vehicle capacity is 3 for each commodity in set 4 and 4 in set 5. All new data are available at <http://www.optimization.dk/MCVRP>.

One of the purposes of this paper is to compare the two strategies, but due to the difficulties of solving the problems with the C-Split strategy, we need relatively small instances for this purpose. We have therefore made the instances with 10–25 customers in set 1, all demanding every commodity. Some customers in set 4 only request one commodity and the algorithm can handle up to 75 customers and 3 commodities reasonable well. For these combinations, we have created 5 instances of each size to provide sufficient instances for our comparison. For all other combinations, we have created 2 instances of each size to favour variation in the test sets.

In our experiments, we also use part of the data of Muyldermans and Pang (2010), which all have 100 customers. As that paper presents a heuristic, many of the instances were too large in terms of capacities for our exact algorithm to tackle. But the instances where the capacity is relatively large compared to the vehicle compartment capacities are included in our tests. The instances originate from three sets, which we refer to as M4, M5, and M6. The 15 instances of set M4 have binary demand, two commodities, and a capacity of three. A number is added to the name of these instances (0, 33, or 66) indicating the percentage of the customers requesting both commodities. The first 10 instances in sets M5 and M6 have demand drawn uniformly between 1 and 5 and compartment capacities of 6 and 12, respectively. These instances are marked with an ‘A’ in our tables. The remaining 25 instances in each of the two sets have demand drawn uniformly between 6 and 10 and compartment capacities between 12 and 36. These instances are marked with an ‘B’ in our tables. The depot is located centrally in sets M4 and M5, and remotely in set M6. All instances in sets M4, M5, and M6 have two commodities.

Table 8 No-Split strategy for binary demand and a capacity of 3

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-3-3-1	1075.9	1044.0	0.9	1053.3	1053.3	*	31	7	10	217
40-3-3-2	1069.5	973.1	2.2	994.5	987.8	0.7	2527	7	10	TL
40-4-3-1	1110.2	1074.0	1.4	1088.9	1087.2	0.2	799	6	9	TL
40-4-3-2	1144.3	1069.8	0.8	1078.0	1078.0	*	125	5	11	204
50-2-3-1	1535.8	1422.7	1.4	1442.7	1436.0	0.5	1528	5	14	TL
50-2-3-2	1351.7	1237.7	0.2	1240.6	1240.6	*	21	5	14	70
50-2-3-3	1434.7	1353.5	2.3	1384.1	1358.0	1.9	1401	5	14	TL
50-2-3-4	1329.4	1286.7	0.7	1296.2	1296.2	*	163	5	14	132
50-2-3-5	1385.9	1300.8	4.8	1363.9	1305.9	4.4	2017	6	13	TL
50-3-3-1	1485.7	1364.9	1.1	1380.5	1373.0	0.5	546	5	12	TL
50-3-3-2	1321.5	1207.7	3.1	1244.5	1217.6	2.2	938	6	12	TL
50-4-3-1	1428.8	1314.6	2.0	1340.7	1316.7	1.8	726	6	11	TL
50-4-3-2	1253.0	1175.1	1.1	1188.6	1186.1	0.2	990	7	13	TL
75-2-3-1	2015.4	1844.7	4.3	1923.5	1847.3	4.1	1087	5	18	TL
75-2-3-2	2059.3	1880.5	0.9	1897.3	1887.2	0.5	369	5	20	TL
75-2-3-3	2142.6	1825.6	4.1	1900.3	1827.8	4.0	631	6	19	TL
75-2-3-4	1954.0	1691.1	2.2	1727.5	1692.4	2.1	192	5	18	TL
75-2-3-5	2157.2	1952.9	4.8	2047.4	1953.8	4.8	341	6	20	TL
75-3-3-1	2218.9	1875.8	3.6	1942.4	1876.9	3.5	656	6	18	TL
75-3-3-2	1924.2	1756.6	3.1	1811.9	1757.4	3.1	180	5	18	TL
75-4-3-1	2188.8	1841.2	3.0	1897.2	1844.6	2.8	415	7	17	TL
75-4-3-2	1875.4	1674.6	0.7	1686.2	1677.7	0.5	104	7	18	TL
100-2-3-1	2867.1	2605.5	1.3	2638.9	2609.4	1.1	686	5	28	TL
100-2-3-2	2761.1	2361.1	0.8	2381.0	2365.5	0.7	579	5	27	TL
100-3-3-1	2803.7	2371.3	3.4	2451.2	2372.5	3.3	162	6	25	TL
100-3-3-2	2603.6	2207.0	2.5	2261.5	2208.2	2.4	328	7	24	TL
100-4-3-1	2820.2	2379.5	2.2	2431.5	2380.0	2.2	205	7	23	TL
100-4-3-2	2592.5	2187.7	2.1	2234.4	2187.7	2.1	101	8	24	TL

5.2 Results for the No-Split strategy

In this section, we consider the results obtained with the No-Split strategy. We first consider Table 2, which reports results for the 44 instances of set 1 having demands between 1 and 5 and vehicle capacity 10. We see, that our algorithm is able to solve most of the instances with up to 50 nodes and that this is often done in less than 10 min. For instances with 75 and 100 nodes, even though we do not prove optimality, the obtained gaps are very small, with a maximum of 3.1%. In total, 28 out of these 44 instances are solved to optimality.

Table 9 No-Split strategy for binary demand and a capacity of 4

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-4-1	1009.7	904.0	7.8	974.3	915.7	6.4	2420	6	9	TL
40-2-4-2	1132.4	969.2	3.6	1004.3	986.7	1.8	1639	5	9	TL
40-3-4-1	984.3	864.8	7.4	928.4	872.6	6.4	453	7	8	TL
40-3-4-2	927.3	853.4	2.3	873.2	854.0	2.2	65	6	7	TL
40-4-4-1	998.6	873.1	0.1	873.6	873.6	*	5	7	8	34
40-4-4-2	971.8	851.9	2.9	876.4	857.5	2.2	137	7	7	TL
50-2-4-1	1203.5	1121.9	1.3	1136.8	1130.7	0.5	331	6	11	TL
50-2-4-2	1207.0	1110.8	8.7	1207.0	1126.4	7.2	1401	6	11	TL
50-3-4-1	1276.5	1095.3	4.6	1145.4	1095.3	4.6	696	6	10	TL
50-3-4-2	1088.0	1025.6	6.1	1088.0	1025.6	6.1	69	7	9	TL
50-4-4-1	1203.4	1089.4	2.4	1115.6	1089.4	2.4	309	6	10	TL
50-4-4-2	1120.1	—	—	1120.1	—	—	0	9	8	TL
75-2-4-1	1684.0	1518.8	4.7	1590.5	1520.6	4.6	671	6	15	TL
75-2-4-2	1721.3	1495.1	2.8	1536.4	1495.4	2.7	248	7	14	TL
75-3-4-1	1865.7	1490.4	3.6	1544.2	1490.4	3.6	465	7	14	TL
75-3-4-2	1629.3	1447.8	3.3	1495.6	1447.8	3.3	133	9	14	TL
75-4-4-1	1582.5	—	—	1582.5	—	—	0	10	12	TL
75-4-4-2	1647.2	—	—	1647.2	—	—	0	8	14	TL
100-2-4-1	2107.4	—	—	2107.4	—	—	0	7	19	TL
100-2-4-2	2366.6	2014.1	5.9	2133.3	2017.6	5.7	69	7	20	TL
100-3-4-1	2106.2	—	—	2106.2	—	—	0	8	17	TL
100-3-4-2	2107.7	—	—	2107.7	—	—	0	8	17	TL
100-4-4-1	2179.4	—	—	2179.4	—	—	0	7	18	TL
100-4-4-2	2241.0	—	—	2241.0	—	—	0	7	19	TL

We now take a look across the three Tables 2, 3, and 4, which report results obtained with the No-Split strategy for instances of sets 1–3, having demand between 1 and 5 and vehicle capacity 10, 15, and 20, respectively. As expected, our ability to solve the problems to optimality decreases when the vehicle capacity increases and with a capacity of 20, we only prove optimality in one instance.

Furthermore, when the capacity increases, there are some instances where even solving the root is too time consuming for our algorithm. For these, *LB* and *Gap* are marked with “—” in Tables 3 and 4, and solving these instances is left as an open challenge for future research. For the remaining instances of Tables 3 and 4, it can be seen that the average gap increases with the vehicle capacity.

The explanation for this is found in the labeling algorithm which creates significantly more labels when capacity is high. This can be seen by noting that the maximum number of customers in the best feasible solution increases from 5 to 8 when the vehicle capacity increases from 10 to 20. As a result, the solution time for

Table 10 No-Split strategy for instances of Fig. 4 of Muyldermans and Pang (2010)

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-3-1-0	2257.9	1945.9	16.0	2257.9	1946.0	16.0	83	6	18	TL
100-2-3-2-0	2244.5	1801.5	24.6	2244.5	1802.1	24.5	9	6	18	TL
100-2-3-3-0	2221.6	1842.3	20.5	2219.3	1843.3	20.4	10	6	18	TL
100-2-3-4-0	1937.0	1714.3	13.0	1937.0	1714.8	13.0	19	6	17	TL
100-2-3-5-0	2227.8	1860.3	19.7	2226.9	1860.4	19.7	3	6	20	TL
100-2-3-1-33	2674.8	2346.66	4.4	2449.9	2349.1	4.3	324	6	24	TL
100-2-3-2-33	2455.1	2237.0	3.1	2307.1	2237.3	3.1	163	6	24	TL
100-2-3-3-33	2534.2	2245.3	3.3	2319.0	2245.3	3.3	1019	6	23	TL
100-2-3-4-33	2540.4	2141.2	3.6	2218.3	2141.5	3.6	497	6	22	TL
100-2-3-5-33	2485.4	2217.2	2.7	2276.8	2218.2	2.6	312	6	23	TL
100-2-3-1-66	2869.4	2748.3	0.6	2765.6	2749.0	0.6	861	4	31	TL
100-2-3-2-66	2956.9	2674.1	2.8	2748.7	2674.7	2.8	218	5	29	TL
100-2-3-3-66	3205.2	2836.6	1.3	2872.8	2840.3	1.1	277	5	30	TL
100-2-3-4-66	2994.4	2593.8	2.2	2649.7	2595.0	2.1	663	5	28	TL
100-2-3-5-66	3056.7	2809.0	0.6	2827.1	2811.0	0.6	119	5	30	TL

each node in the branching tree increases and the number of nodes that can be investigated within the time limit decreases.

We see a similar pattern when we turn our attention to Tables 5 and 6 that present the results for the M5 and M6 sets of Muyldermans and Pang (2010) with similar type of demand. The instances with relatively few customers on each route are solved to optimality within a limited time, whereas the time limit is reached when compartment capacities increase. We observe no significant difference in computation times across the two sets. It is worth noting, that for instances 11 through 15 in each of these tables, the combination of demand and compartment capacities is such that the optimal solutions almost uses separate vehicles for each customer.

Tables 7, 8, and 9 show the results obtained with the No-Split strategy for instances of sets 4 and 5, having binary demand and vehicle capacities 3 and 4, respectively. The results for the case of a vehicle capacity of 3 are very similar to those seen in Table 2 as we are able to solve instances with up to 50 nodes to optimality within a short time. However, not all instances with 40 and 50 nodes could be solved within the time limit. 48 out of the 73 instances in set 4 were solved to optimality. Note that due to the nature of the binary data, there are up to 8 customers on a route even though the vehicle capacity is only 3 because all commodities are not ordered by all customers. When we look at instances with vehicle capacity 4 as shown in Table 9, we see that only one instance is solved to optimality and the average gap increases from 0.7 to 3.7% with a maximum gap of about 7.2% for the hardest solvable instances with 100 customers and 2 commodities. However, these numbers do not reflect the fact that 8 instances in

Table 11 C-Split strategy for U(1, 5) demand and a capacity of 10

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-10-1	524.1	495.8	1.5	503.3	503.3	*	11	6	4	7
10-2-10-2	477.6	410.2	*	410.2	410.2	*	0	8	3	117
10-2-10-3	461.9	461.2	*	461.2	461.2	*	0	6	4	5
10-2-10-4	463.8	435.6	3.7	451.7	450.3	0.3	63	8	4	TL
10-2-10-5	477.0	392.7	0.9	396.1	396.1	*	35	8	3	30
15-2-10-1	737.4	616.4	*	616.4	616.4	*	0	7	6	9
15-2-10-2	662.3	564.9	*	580.1	580.1	*	0	8	5	14
15-2-10-3	553.1	479.6	5.5	501.3	501.3	*	736	6	5	835
15-2-10-4	675.3	640.7	2.5	656.9	656.9	*	513	6	7	419
15-2-10-5	619.1	520.7	0.8	524.7	524.7	*	134	7	5	1477
20-2-10-1	924.7	833.6	2.0	843.8	843.8	*	258	7	8	946
20-2-10-2	692.6	613.7	2.4	628.5	628.5	*	69	8	6	350
20-2-10-3	668.3	636.8	4.6	666.1	643.7	3.5	81	9	7	TL
20-2-10-4	943.3	876.6	0.8	882.3	882.3	*	141	7	8	981
20-2-10-5	771.0	670.2	2.1	684.0	671.7	1.8	283	8	7	TL
25-2-10-1	1049.7	967.8	0.8	972.2	972.2	*	59	9	8	3206
25-2-10-2	821.9	733.2	3.0	744.8	736.7	2.5	86	8	7	TL
25-2-10-3	887.3	824.7	0.1	825.9	825.3	0.1	20	10	9	TL
25-2-10-4	1143.7	1026.5	2.1	1047.8	1031.9	1.5	127	7	10	TL
25-2-10-5	975.8	827.4	3.0	852.3	833.2	2.3	56	8	9	TL

set 5 are left with an unsolved root node after the time limit is reached. For these instances, we marked *LB* and *Gap* with “—” in Table 9. Table 10 presents the results of set M4 with the same type of demand. We note that these instances seem easier than the instances with 100 customers in set 4.

5.3 Results for the C-Split strategy

The results for the C-Split strategy are shown in Tables 11, 12, 13, 14 and 15 and it is evident that solving this problem is harder. Because the C-Split is harder to solve, we only present results for instances up to the size that could be handled by the algorithm. For set 1 (Table 11), with U(1, 5) demand with capacity 10, even some instances with 25 customers could not be solved within the time limit whereas the same instances were solved in less than 2 min with the No-Split strategy. Tables 12 and 13, showing results for M5 and M6, indicate a similar pattern. Out of the total of 30 instances in these tables, only one could be solved to optimality within an hour, whereas with the No-Split strategy, all 30 instances could be solved to optimality. Tables 14 and 15 show the C-Split strategy for the binary data with vehicle capacity of 3 and when we compare these tables to Tables 7 and 8, we see the same general

Table 12 C-Split strategy for instances of Fig. 5 of Muyldermans and Pang (2010)

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	5668.8	4618.6	22.7	5668.8	4618.6	22.7	14	5	54	TL
100-2-6-2-A	5805.7	4760.3	0.2	4768.5	4761.4	0.2	9	5	58	TL
100-2-6-3-A	5818.1	4503.6	29.2	5818.1	4503.7	29.2	2	6	58	TL
100-2-6-4-A	5841.7	4717.6	23.8	5841.7	4717.6	23.8	2	5	57	TL
100-2-6-5-A	5583.5	4562.5	22.4	5583.5	4562.6	22.4	24	6	52	TL
100-2-12-1-B	7812.8	7501.4	0.1	7510.7	7503.1	0.1	352	4	94	TL
100-2-12-2-B	7561.0	6912.1	0.1	6921.3	6917.4	0.1	315	4	92	TL
100-2-12-3-B	7409.8	6908.7	0.1	6917.3	6911.3	0.1	171	4	91	TL
100-2-12-4-B	7989.5	7422.0	0.4	7448.5	7428.3	0.3	209	4	91	TL
100-2-12-5-B	7868.9	7245.8	0.3	7270.5	7254.1	0.2	250	4	92	TL
100-2-18-1-B	4853.8	4295.4	13.0	4853.8	4297.2	13.0	13	5	51	TL
100-2-18-2-B	4482.6	3969.2	12.9	4482.6	3969.2	12.9	12	5	50	TL
100-2-18-3-B	4803.5	4061.6	18.3	4803.5	4062.0	18.3	18	5	51	TL
100-2-18-4-B	4834.5	4246.0	0.2	4255.5	4246.0	0.2	15	6	49	TL
100-2-18-5-B	5057.4	4155.6	21.7	5057.4	4155.6	21.7	13	5	51	TL

Table 13 C-Split strategy for instances of Fig. 6 of Muyldermans and Pang (2010)

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
100-2-6-1-A	11644.3	10784.3	8.0	11644.3	10784.3	8.0	19	6	51	TL
100-2-6-2-A	12936.6	12066.6	7.2	12930.8	12066.7	7.2	7	7	57	TL
100-2-6-3-A	12452.6	11246.2	10.7	12452.6	11246.3	10.7	47	5	54	TL
100-2-6-4-A	12190.7	10747.4	13.2	12190.7	10747.4	13.4	5	7	54	TL
100-2-6-5-A	13490.0	12459.3	8.2	13476.1	12459.3	8.2	24	5	61	TL
100-2-12-1-B	20518.8	20206.6	0.0	20206.6	20206.6	*	0	4	94	292
100-2-12-2-B	19802.2	19452.2	0.5	19543.6	19455.7	0.5	177	4	91	TL
100-2-12-3-B	19713.3	19454.8	0.3	19522.7	19455.8	0.3	182	4	90	TL
100-2-12-4-B	20196.1	19875.7	0.3	19943.3	19878.9	0.3	3	3	94	TL
100-2-12-5-B	19542.6	19171.2	0.4	19256.3	19174.6	0.4	204	3	91	TL
100-2-18-1-B	11332.5	10616.4	6.7	11332.5	10616.4	6.7	26	5	51	TL
100-2-18-2-B	11023.6	10305.4	7.0	11023.6	10305.4	7.0	11	5	50	TL
100-2-18-3-B	11112.7	10337.4	7.5	11112.7	10337.4	7.5	25	5	50	TL
100-2-18-4-B	11223.7	10472.1	7.2	11223.7	10472.1	7.1	11	5	51	TL
100-2-18-5-B	10938.8	10212.3	7.1	10937.8	10212.3	7.1	38	5	50	TL

Table 14 C-Split strategy for binary demand and a capacity of 3

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
10-2-3-1	411.5	398.9	3.0	410.9	410.9	*	21	5	3	6
10-2-3-2	458.9	424.2	1.7	431.6	431.6	*	3	5	4	4
10-2-3-3	357.0	350.2	1.9	357.0	357.0	*	3	5	3	2
10-2-3-4	441.9	393.2	7.2	421.4	421.4	*	290	6	3	84
10-2-3-5	390.6	386.9	*	386.9	386.9	*	0	6	3	1
10-3-3-1	389.8	375.8	3.7	389.8	389.8	*	3	9	2	5
10-3-3-2	397.3	372.3	1.0	376.0	376.0	*	3	9	3	2
10-3-3-3	460.6	393.2	7.2	421.4	415.2	1.5	7320	9	3	TL
10-3-3-4	427.1	392.9	0.8	396.2	396.2	*	147	9	3	141
10-3-3-5	422.3	396.5	*	396.5	396.5	*	0	8	3	4
15-2-3-1	530.7	507.3	3.3	523.8	523.8	*	989	6	4	290
15-2-3-2	536.3	508.5	2.1	518.9	518.9	*	3	6	5	3
15-2-3-3	556.4	507.0	3.4	524.1	524.1	*	59	6	5	22
15-2-3-4	462.4	444.4	2.0	453.2	453.2	*	17	6	4	7
15-2-3-5	554.7	526.8	4.6	550.9	550.9	*	455	6	4	372
15-3-3-1	470.9	460.8	1.2	466.3	466.3	*	3	9	4	17
15-3-3-2	532.1	479.4	1.6	487.5	479.4	1.7	42	8	4	TL
15-3-3-3	526.9	478.5	2.5	490.4	481.6	1.8	346	8	4	TL
15-3-3-4	470.5	444.7	1.4	451.0	451.0	*	37	9	4	358
15-3-3-5	415.9	380.1	3.5	393.4	382.7	2.8	483	9	4	TL
20-2-3-1	687.8	650.1	0.3	651.9	651.9	*	5	6	5	21
20-2-3-2	606.1	588.6	1.7	598.6	595.3	0.6	21	6	6	TL
20-2-3-3	702.6	690.8	1.3	700.1	690.8	1.4	263	6	6	TL
20-2-3-4	600.5	579.5	0.4	581.9	581.9	*	55	5	6	38
20-2-3-5	592.6	556.6	1.6	565.6	565.6	*	301	5	6	498
20-3-3-1	632.5	592.7	3.1	611.2	595.0	2.7	89	9	5	TL
20-3-3-2	601.0	551.4	2.3	564.1	554.4	1.7	545	9	5	TL
20-3-3-3	695.2	617.4	2.8	634.4	617.4	2.7	293	9	6	TL
20-3-3-4	670.3	562.0	1.9	572.4	564.0	1.5	70	9	5	TL
20-3-3-5	700.7	562.6	4.3	587.0	566.8	3.6	155	9	5	TL
30-2-3-1	972.8	841.4	0.1	842.3	842.3	*	5	6	9	36
30-2-3-2	942.4	829.4	2.0	845.9	833.3	1.5	296	6	9	TL
30-2-3-3	1199.8	985.9	1.8	1003.5	988.3	1.5	549	6	9	TL
30-2-3-4	946.5	829.4	2.1	849.6	831.7	2.2	197	6	9	TL
30-2-3-5	850.3	784.9	1.6	797.5	791.5	0.8	1207	6	8	TL
30-3-3-1	1014.6	870.2	6.5	927.0	870.2	6.5	41	9	7	TL
30-3-3-2	1022.9	785.6	4.1	817.9	785.6	4.1	143	9	8	TL
30-3-3-3	1136.1	937.1	2.5	960.2	937.1	2.5	26	9	9	TL
30-3-3-4	848.3	797.5	1.6	809.9	797.5	1.6	110	9	8	TL
30-3-3-5	1095.3	869.0	3.4	899.0	871.2	3.2	317	9	9	TL

Table 14 continued

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
40-2-3-1	1211.3	1120.7	0.6	1127.0	1127.0	*	75	6	10	267
40-2-3-2	1218.4	1055.2	2.0	1076.1	1056.1	1.9	429	6	11	TL
40-2-3-3	1268.6	1136.0	2.5	1165.0	1140.6	2.1	721	6	11	TL
40-2-3-4	1350.3	1113.2	1.9	1134.0	1119.3	1.3	265	6	12	TL
40-2-3-5	1237.2	1049.3	1.6	1065.9	1049.4	1.6	349	6	11	TL

Table 15 C-Split strategy for binary demand and a capacity of 3

ID	Root			Best known			Info			
	SA	LB	Gap	UB	LB	Gap	Nodes	Cust	Veh	Time
50-2-3-1	1589.7	1409.0	1.5	1429.8	1411.6	1.3	340	6	14	TL
50-2-3-2	1339.0	1229.1	0.1	1229.8	1229.8	*	23	6	14	854
50-2-3-3	1529.5	1341.4	3.2	1383.8	1341.4	3.2	48	6	14	TL
50-2-3-4	1458.7	1280.5	1.9	1305.2	1281.9	1.8	44	6	14	TL
50-2-3-5	1494.2	1295.9	3.4	1340.4	1296.0	3.4	156	6	13	TL
75-2-3-1	2183.4	1837.3	5.7	1942.2	1838.2	5.7	290	6	18	TL
75-2-3-2	2130.0	1860.0	4.8	1949.7	1861.0	4.8	186	6	20	TL
75-2-3-3	2040.8	1803.7	5.3	1899.8	1805.1	5.3	302	6	18	TL
75-2-3-4	2009.3	1682.1	3.8	1745.6	1682.8	3.7	257	6	17	TL
75-2-3-5	2299.5	1945.4	5.1	2044.7	1945.4	5.1	186	6	20	TL

picture of the C-Split being harder for our algorithm to solve than the No-Split as we have seen for the other sets. For this set, we can solve about half on the instances to optimality. For both sets 1 and 4, the C-Split cannot consistently solve problems with more than 20 nodes within an hour.

The explanation for this is to be found in the labeling algorithm for the two strategies. When allowing C-Split, the labeling algorithm will treat every commodity for every customer as a separate customer because any combination of including or excluding a commodity in the route is possible. Therefore, for the C-Split, the labeling algorithm will be burdened by this high number of artificial customers. Note that for the C-Split strategy, the *Cust* column reports the number of these artificial customers rather than the number of physical customers.

5.4 Comparison of the No-Split and the C-Split strategies

In this section, we compare the costs resulting from each of the two strategies in order to determine whether companies can obtain savings in transportation cost by applying a C-Split strategy rather than a No-Split strategy. We therefore consider the 105 instances that were solvable by both algorithms and summarize our findings in Table 16. The table shows the results for the full set of 105 instances and for the four sets individually.

Out of the 105 instances, 37 were solved to optimality for both strategies and based on the results obtained for those 37 instances we can conclude that (1) the two strategies led to the same objective value for 25 of the instances (24%), thus allowing C-Split for these instances did not result in cost savings; and (2) for the remaining 12 instances (11.4%) cost savings were obtained by allowing C-Split at an average rate of 1.7%.

As regards the remaining 68 instances, we note that if an instance is solved to optimality for the No-Split problem but not for C-Split, and if the best solution obtained for C-Split is lower than the optimal solution for No-Split, then there will be a certain cost saving by applying C-Split. There are 27 such instances and the average cost saving for these instances is 2.9%. We point out that the cost saving is under-estimated because the optimal C-Split value may be lower than the best known. We cannot make any conclusions as to possible benefits from C-Split based on the obtained values for the remaining 41 instances.

In summary: for 24% of the instances, the solution for the two strategies are the same, for 37% of the instances there is a some cost saving, and for the remaining 39%, no conclusion can be drawn from our analysis.

Among instances where we obtain a certain saving, the instances in set M6 generally result in the highest savings. The explanation for this is likely to be found in the remote location of the depot which means that a significant travel distance can be saved if C-Split results in a decrease in the number of vehicles used. However, it should be noted that for the M6 instances the demand is such that most customers are serviced by a separate vehicle in the No-Split scenario and hence, these instances are not representative.

Table 16 Comparison of C-Split to No-Split

	Set 1	Set 4	Set M5	Set M6	Total
Number of instances	20	55	15	15	105
Number: both strategies solved opt	13	22	0	1	37
Number: $OPT_{C-Split} = OPT_{No-Split}$	7	17	0	0	25
$OPT_{C-Split} < OPT_{No-Split}$	6	5	0	1	12
Average saving, %	1.8	1.0	0	4.0	1.7
Number: $Best_{C-Split} < OPT_{No-Split}$	4	12	7	4	27
Average saving, %	1.3	1.0	2.6	6.6	2.9
Number: $Best_{C-Split} \geq ? OPT_{No-Split}$	3	21	8	10	41

We have performed additional tests on 11 instances of set 1 with modified demand using an objective of minimizing the number of vehicles used. These tests, which are not shown here, indicate that it is generally not possible to reduce the number of vehicles by allowing C-Split. However, additional analysis is needed in order to draw more precise conclusions.

6 Concluding remarks

In this paper, we have presented an exact algorithm based on Branch-and-Price for solving two versions of the Multi-Commodity Vehicle Routing Problem, one that allows different commodities to a customer to be delivered by different vehicles (C-Split) and one that does not allow such splitting (No-Split). The performance of our algorithm is analyzed based on a total of 274 instances. We are able to solve instances with up to 50 nodes and 4 commodities for No-Split. 108 of the instances have been solved to optimality for the No-Split strategy. Our performance limit is slightly lower as regards the more complex C-Split problem; we analyzed 105 instances and solved 37 of them to optimality.

Our comparison of the two strategies is based on 105 instances. We found that routing cost benefits in 39 of these instances by using commodity splitting, but in 25 instances, splitting did not lead to cost savings. As for the remaining 41 instances we were not able to draw a conclusion.

For instances that did get a cost saving from the rather more complicated C-Split strategy as opposed to the No-Split strategy, the saving rate averaged 2.3%. It would be interesting to see whether the advantage of using C-Split persists for large instances.

References

- Archetti C, Bianchessi N, Speranza MG (2015) A branch-price-and-cut algorithm for the commodity constrained split delivery vehicle routing problem. *Comput Oper Res* 64:1–10
- Archetti C, Campbell AM, Speranza MG (2014) Multicommodity vs. single-commodity routing. *Transp Sci* 50(2):461–472
- Avella P, Boccia M, Sforza A (2004) Solving a fuel delivery problem by heuristic and exact approaches. *Eur J Oper Res* 152(1):170–179
- Ben Abdelaziz F, Roucairol C, Bacha C (2002) Deliveries of liquid fuels to snbp gas stations using vehicles with multiple compartments. In: 2002 IEEE International Conference on Systems, Man and Cybernetics, vol 1, pp 478–483
- Boland N, Dethridge J, Dumitrescu I (2006) Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Oper Res Lett* 34(1):58–68
- Brown GG, Ellis CJ, Graves GW, Ronen D (1987) Real-time, wide area dispatch of mobil tank trucks. *Interfaces* 17(1):107–120
- Brown GG, Graves GW (1981) Real-time dispatch of petroleum tank trucks. *Manag Sci* 27(1):19–32
- Chajakis ED, Guignard M (2003) Scheduling deliveries in vehicles with multiple compartments. *J Glob Optim* 26:43–78
- Coelho LC, Laporte G (2015) Classification, models and exact algorithms for multi-compartment delivery problems. *Eur J Oper Res* 242(3):854–864

- Cornillier F, Boctor FF, Laporte G, Renaud J (2008) An exact algorithm for the petrol station replenishment problem. *J Oper Res Soc* 59(5):607–615
- Cornillier F, Boctor FF, Laporte G, Renaud J (2008) A heuristic for the multi-period petrol station replenishment problem. *Eur J Oper Res* 191(2):295–305
- Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. *Oper Res* 8(1):101–111
- Dantzig GB, Wolfe P (1961) The decomposition algorithm for linear programs. *Econometrica* 29(4):767–778
- Derigs U, Gottlieb J, Kalkoff J, Piesche M, Rothlauf F, Vogel U (2011) Vehicle routing with compartments: applications, modelling and heuristics. *OR Spectr* 33(4):885–914
- Desaulniers G, Desrosiers J, Solomon M (eds.) Column generation. Springer (2005)
- El Fallahi A, Prins C, Calvo RW (2008) A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. *Comput Oper Res* 35(5):1725–1741
- Henke T, Speranza MG, Wäscher G (2015) The multi-compartment vehicle routing problem with flexible compartment sizes. *Eur J Oper Res* 246(3):730–743
- Irnich S, Desaulniers G (2005) Shortest path problems with resource constraints. In: Desaulniers G, Desrosiers J, Solomon M (eds.) Column generation. Springer, pp 33–65
- Lahyani R, Coelho LC, Khemakhem M, Laporte G, Semet F (2015) A multi-compartment vehicle routing problem arising in the collection of olive oil in Tunisia. *Omega* 51:1–10
- Laporte G, Gendreau M, Potvin JY, Semet F (2000) Classical and modern heuristics for the vehicle routing problem. *Int Trans Oper Res* 7(4–5):285–300
- Muyldermans L, Pang G (2010) On the benefits of co-collection: experiments with a multi-compartment vehicle routing algorithm. *Eur J Oper Res* 206(1):93–103
- Reed M, Yiannakou A, Evering R (2014) An ant colony algorithm for the multi-compartment vehicle routing problem. *Appl Soft Comput* 15:169–176
- Repoussis PP, Tarantilis CD, Ioannou G (2007) A hybrid metaheuristic for a real life vehicle routing problem. In: Numerical methods and applications. Springer, pp 247–254
- Ryan DM, Foster BA (1981) An integer programming approach to scheduling. In: Wren A (ed) Computer scheduling of public transport urban passenger vehicle and crew scheduling. North-Holland, Amsterdam, pp 269–280
- Toth P, Vigo D (2014) Vehicle routing: Problems, methods, and applications. MOS-SIAM Series on optimization
- Vanderbeck F (2000) On Dantzig–Wolfe decomposition in integer programming and ways to perform branching in a branch-and-price algorithm. *Oper Res* 48(1):111–128
- Van der Bruggen L, Gruson R, Salomon M (1995) Reconsidering the distribution structure of gasoline products for a large oil company. *Eur J Oper Res* 81(3):460–473
- Wang Q, Ji Q, Chiu CH (2014) Optimal routing for heterogeneous fixed fleets of multicompartment vehicles. *Math Probl Eng*