The Periodic Vehicle Routing Problem with Driver Consistency

Inmaculada Rodríguez-Martín*, Juan-José Salazar-González

DMEIO, Facultad de Ciencias, Universidad de La Laguna, Tenerife, Spain

irguez@ull.es , jjsalaza@ull.es

Hande Yaman

Department of Industrial Engineering, Bilkent University, Ankara, Turkey

hyaman@bilkent.edu.tr

Abstract

The Periodic Vehicle Routing Problem is a generalization of the classical capacitated vehicle routing problem in which routes are determined for a planning horizon of several days. Each customer has an associated set of allowable visit schedules, and the objective of the problem is to design a set of minimum cost routes that give service to all the customers respecting their visit requirements. In this paper we study a new variant of this problem in which we impose that each customer should be served by the same vehicle/driver at all visits. We call this problem the Periodic Vehicle Routing Problem with Driver Consistency. We present an integer linear programming formulation for the problem and derive several families of valid inequalities. We solve it using an exact branch-and-cut algorithm, and show computational results on a wide range of randomly generated instances.

Keywords: routing, periodic vehicle routing, driver consistency, valid inequalities, branch-and-cut

1 Introduction

The *Periodic Vehicle Routing Problem* (PRVP), introduced by Beltrami and Bodin [6], is a generalization of the classical capacitated vehicle routing problem (VRP) in which routes are determined for a planning horizon of multiple periods with some customers demanding multiple visits. There are several possible schedules for visiting each customer. For instance, if we are making a plan for Monday to Friday and if a customer needs to be visited twice with

*Corresponding author

at least one day and at most two days between consecutive visits, then Monday - Wednesday, Monday - Thursday, Tuesday - Thursday, Tuesday - Friday, and Wednesday - Friday are possible visit schedules for this customer. The problem is to decide on the schedules and the routes simultaneously to minimize the total transportation cost.

In [6], this problem has been studied for municipal waste collection and a heuristic approach is proposed. Since this seminal work, many heuristic approaches have been proposed to solve the PVRP and its variants. See, for instance, [11, 12, 15, 16, 19, 25, 36, 37, 40]. On the other hand, exact methods for the PVRP and its variants are rare. Mourgaya and Vanderbeck [33] propose a column generation for the problem of determining the visit schedules and customer assignments to vehicles without considering the sequencing of customers on routes. Butler et al. [9] present a branch-and-cut algorithm for a two-period travelling salesman problem. Francis et al. [17] study a variant in which frequencies of visits are also decisions. They propose a solution approach based on Lagrangian relaxation and branch-and-bound. Baldacci et al. [3] propose an exact algorithm for the PVRP that solves a route based formulation by first computing a near optimal dual solution and then using the dual information to restrict the set of routes without losing the optimal solution.

Among the applications of the PVRP are the planning of deliveries of hospital linen to clinics [5], visits for preventive maintenance or quality assurance [7, 23], delivery of blood products to hospitals [24], visits to suppliers or customers in a supply chain [1, 13, 20, 21, 29, 35], visits to collect recyclable materials and waste [4, 8, 14, 34, 38, 39, 41], routes for lottery sales [26], or visits to students or patients at home [2, 32].

For more details on the applications, solution methods and variants of the PVRP, we refer the reader to the surveys by Campbell and Wilson [10], and by Francis et al. [18].

Groër et al. [22] introduce the *Consistent VRP*, in which each customer should be visited by the same vehicle and around the same time of the day. Different from the PVRP, there is a unique visit schedule for each customer and it is known in advance. Zhu et al. [42] and Luo et al. [31] study a multi period vehicle routing problem with time windows in which a customer can be served by at most a certain number of different vehicles over the planning horizon. Kovacs et al. [27] also study a generalization which allows a certain number of vehicles to visit a node and penalizes the time inconsistency in the objective function instead of enforcing it through constraints.

In this study, we introduce the *Periodic Vehicle Routing Problem with Driver Consistency* (PVRP-DC), in which we restrict the PVRP by imposing that each customer should be visited by the same vehicle at all visits. This problem is motivated by an application from a soft drinks company that collects the demand by visiting its customers regularly. The number of visits to a customer per week is one, two or three depending on the sales volume, and all visits are performed by the same employee. A similar problem is encountered in a delivery system where vehicles visit retailers to take orders and deliver the items simultaneously. In this system the drivers have knowledge about the demand at the retailers and decide on how to load their vehicles with different items based on their forecasts. It is crucial that a retailer is visited by the same driver at each visit for the learning of the driver. These are two examples from the industry where driver consistency is required. A third industrial example is routing of merchandisers who visit chain supermarkets to refill the empty shelves, to better display the products and promotions. It is again critical that the same merchandiser visits the same supermarket as the knowledge of the supermarket and the customer profile is very important. There are also home care applications, like regular home visits by professional caregivers for elderly people or people with special needs. The contact between the caregiver and the person visited is of critical importance in these applications. In modeling these examples, we consider customers with unit demand, as visits have more or less equal durations, and we limit the number of customers that can be visited by a driver in a day to ensure quality of service.

An important issue to notice is that the consistency requirement really has an effect on the solutions of the PVRP, as can be seen in Figures 1 and 2. These figures illustrate the optimal solutions of the PVRP and the PVRP-DC on an instance with ten customers (nodes 1 to 10), depot located at node 0, a time horizon of two periods or days, and two vehicles that can visit at most four customers each. Customers 1, 2, 4, 5, 8 and 9 have to be visited both days. Customer 10 must be visited exclusively the first day, and customer 7 must be visited the second day only. Customers 3 and 6 have to be visited once, either the first or the second day. The optimal PVRP solution for this instance, depicted in Figure 1, has a cost equal to 688.64. To distinguish the routes of the two vehicles they are represented by dashed or solid lines. We can see that customer 1 is visited by a vehicle the first day, and by a different one the second day. If we ask for driver consistency the optimal solution is the one depicted in Figure 2. In this solution all customers are always visited by the same driver, though, as expected, this results in a cost increase.

The PVRP-DC belongs to the class of consistent vehicle routing problems where the focus is on driver consistency rather than visit time consistency. In this article our aim is to derive a strong formulation and valid inequalities for PVRP-DC, and to use those results to propose an exact solution approach to tackle the problem. We restrict our attention to the unit demands case, since most motivating applications are based on delivering service rather than products.

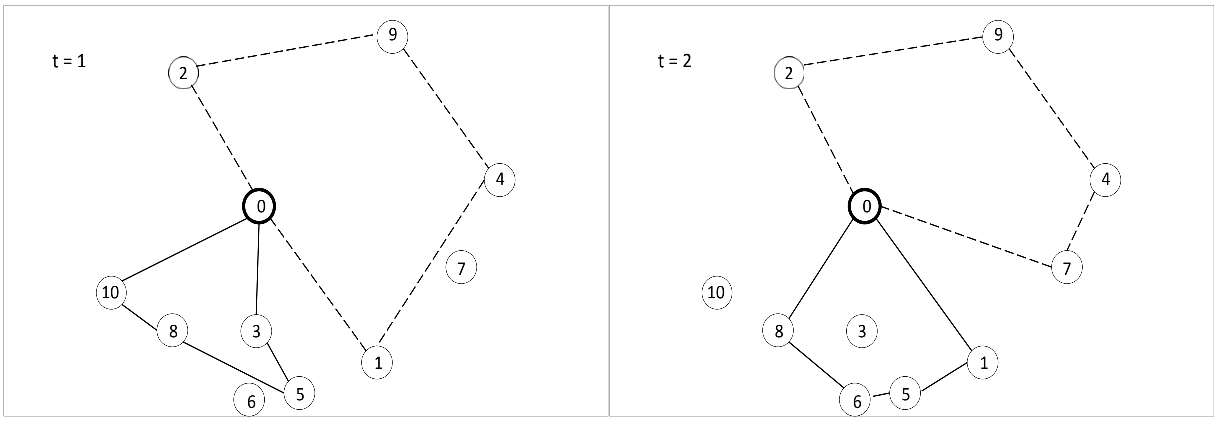

Figure 1: A PVRP solution example (cost = 688.64)

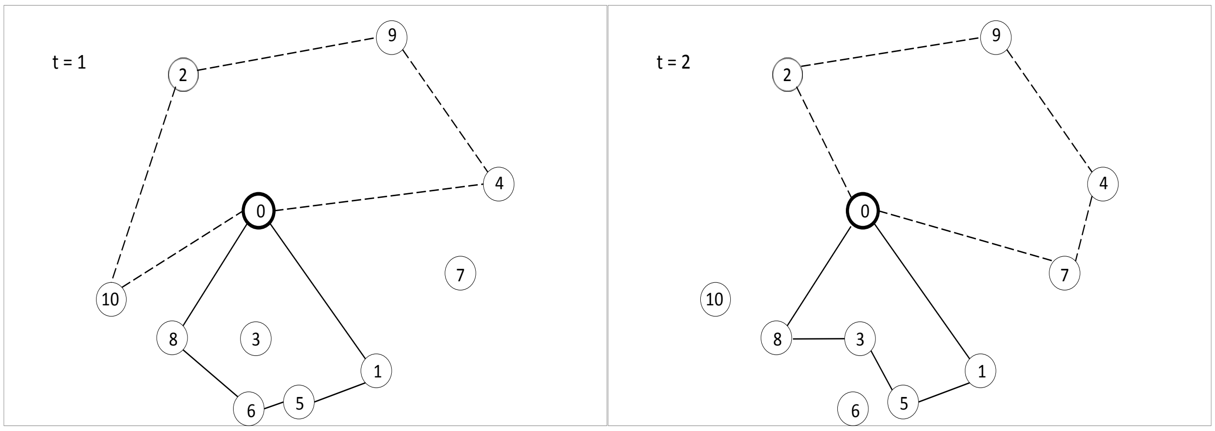

Figure 2: A PVRP-DC solution example (cost = 697.95)

We want to study the effects of combining periodicity and driver consistency, this being the main contribution of our work. Nevertheless, our model and valid inequalities can be easily extended to the general demands case. The remainder of the paper is organized as follows. The problem is formally defined and modeled in Section 2. In Section 3 we present several families of valid inequalities to strengthen the linear programming (LP) relaxation of the model. We explain the details of our branch-and-cut algorithm in Section 4. The computational results are presented and discussed in Section 5. And finally, the paper ends with conclusions in Section 6.

2 MIP Formulation

In this section we give a formal definition of the PVRP-DC and present an integer programming formulation.

Let $V = \{0, 1, \ldots, n\}$ be a set of nodes, with node 0 corresponding to the depot and the other nodes corresponding to customers. Let $E = \{e \subset V : |e| = 2\}$ be the set of edges and c_e denote the transportation cost associated with edge $e \in E$. We consider a planning horizon $T = \{1, \ldots, \tau\}$ of τ periods. Each customer needs to be visited a certain number of times during the planning horizon, always by the same vehicle. We define P_i to be the set of possible visit schedules for customer $i \in V \setminus \{0\}$. A homogenous fleet $K = \{1, \ldots, m\}$ of m vehicles is available at the depot, and each vehicle can visit between 2 and q customers in each period.

The aim of the PVRP-DC is to choose a visit schedule for each customer and to design the routes for each period in order to minimize the total transportation cost over the planning horizon.

We use the following binary decision variables to formulate the problem. We define z_{ip}^k to be 1 if schedule $p \in P_i$ is chosen to serve customer $i \in V \setminus \{0\}$ and if all visits in that schedule are done by vehicle $k \in K$, and 0 otherwise. We also define x_e^{tk} to be 1 if edge $e \in E$ is traversed by vehicle $k \in K$ in period $t \in T$ and 0 otherwise, and y_0^{tk} to be 1 if vehicle $k \in K$ is used in period $t \in T$ and 0 otherwise. To simplify the notation, we let $y_i^{tk} = \sum_{p \in P_i : t \in p} z_{ip}^k$ for $i \in V \setminus \{0\}$, $t \in T$ and $k \in K$. This variable is 1 if customer i is visited by vehicle k in period t and is 0 otherwise.

We use some additional notation. For $S \subseteq V$, let $\delta(S) = \{e \in E : |S \cap e| = 1\}$. If $S = \{i\}$, we write $\delta(i)$ instead of $\delta(\{i\})$. In addition, for a given subset of edges $E' \subseteq E$, we define $x^{tk}(E') = \sum_{e \in E'} x_e^{tk}$.

The PVRP-DC can be modeled as follows:

$$\min \sum_{t \in T} \sum_{k \in K} \sum_{e \in E} c_e x_e^{tk} \tag{1}$$

$$\text{s.t.} \sum_{p \in P_i} \sum_{k \in K} z_{ip}^k = 1 \qquad i \in V \setminus \{0\}, \tag{2}$$

$$y_i^{tk} = \sum_{p \in P_i : t \in p} z_{ip}^k \qquad i \in V \setminus \{0\}, k \in K, t \in T, \tag{3}$$

$$x^{tk}(\delta(i)) = 2 y_i^{tk} \qquad i \in V, k \in K, t \in T, \tag{4}$$

$$x^{tk}(\delta(S)) \geq 2 y_i^{tk} \qquad S \subseteq V \setminus \{0\}, i \in S, k \in K, t \in T, \tag{5}$$

$$\sum_{i \in V \setminus \{0\}} y_i^{tk} \leq q y_0^{tk} \qquad k \in K, t \in T, \tag{6}$$

$$x_e^{tk} \in \{0, 1\} \qquad e \in E, k \in K, t \in T, \tag{7}$$

$$y_i^{tk} \in \{0, 1\} \qquad i \in V, k \in K, t \in T, \tag{8}$$

$$z_{ip}^k \in \{0, 1\} \qquad i \in V \setminus \{0\}, p \in P_i, k \in K. \tag{9}$$

The objective function (1) is to minimize the total routing cost. Constraints (2) ensure that each customer is serviced by one vehicle, and following one of its allowable visit schedules. Variables y and z are related through constraints (3). Constraints (4) are the degree constraints for the depot and the customers.

Constraints (5) ensure the connectivity of the vehicle routes. They state that if a customer $i \in S$ is visited by a vehicle k in period t, then the cut $\delta(S)$ must be crossed at least twice by vehicle k in this time period. When $S = V \setminus \{0\}$, for any given $i \in S$ constraints (5) can be written as $x^{tk}(\delta(0)) \geq 2y_i^{tk}$, which becomes, using the degree equation (4) for the depot,

$$y_i^{tk} \leq y_0^{tk} \qquad i \in V \setminus \{0\}, k \in K, t \in T. \tag{10}$$

These inequalities avoid visiting customer i by vehicle k in period t if vehicle k is not used in that period.

Constraints (6) are capacity constraints, and limit to at most q the number of customers that a vehicle k can visit in a period t. They also avoid visiting a node by a vehicle k in period t if this vehicle is not used in period t. Finally, constraints (7)-(9) are variable restrictions.

3 Valid inequalities

In this section we present several families of valid inequalities to strengthen the LP relaxation of the PVRP-DC. The effectiveness of these inequalities will be discussed later, in the section devoted to computational results. Let X be the set of feasible solutions of the model presented above.

The first family of valid inequalities is commonly used in solving similar vehicle routing problems; hence we present it without proof. The family is:

$$x_{0i}^{tk} \leq y_i^{tk} \qquad i \in V \setminus \{0\}, k \in K, t \in T. \tag{11}$$

These inequalities ensure that, if an edge adjacent to the depot is traversed by vehicle k in period t, then its other endpoint is visited by vehicle k in the same period t.

The second family is inspired on the generalized multistar inequalities (see Letchford et al. [30]).

Proposition 1 *Let $S \subseteq V \setminus \{0\}$, $k \in K$, and $t \in T$. The inequality*

$$x^{tk}(\delta(S)) \geq 2\frac{\sum_{i\in S} y_i^{tk} + \sum_{i \in V\setminus(S\cup\{0\})} x^{tk}(E(i:S))}{q} \tag{12}$$

with $E(i:S) = \{e \in E : i \in e, |e \cap S| \neq \emptyset\}$ is a valid inequality for set X.

Proof. The left hand side of inequality (12) is the number of times vehicle k enters and leaves set S in period t. It is evident that $2\sum_{i\in S} y_i^{tk}/q$ is a lower bound for this number, since $\sum_{i\in S} y_i^{tk}$ is the number of customers in S that must be served by vehicle k in period t. But note that this can be increased by also considering all customers served by k in period t that are visited immediately before or after having visited a customer $i \in S$. Hence, (12) is valid. □

The next two families of valid inequalities are specifically derived for the PVRP-DC.

Proposition 2 *Let $T' \subseteq T$, $K' \subseteq K$ and $S \subseteq V \setminus \{0\}$. The inequality*

$$\sum_{t\in T'}\sum_{k\in K'} x^{tk}(\delta(S)) \geq 2\left(\left\lceil\frac{|S|}{q}\right\rceil - \sum_{i\in S}\sum_{k\in K'}\sum_{p\in P_i:p\cap T'=\emptyset} z_{ip}^k - \sum_{i\in S}\sum_{k\in K\setminus K'}\sum_{p\in P_i} z_{ip}^k\right) \tag{13}$$

is a valid inequality for set X.

Proof. This inequality is valid since the number of nodes of set S that are assigned to vehicles in K' to be served during a period in T' is equal to $|S| - \sum_{i\in S}\sum_{k\in K'}\sum_{p\in P_i:p\cap T'=\emptyset} z_{ip}^k - \sum_{i\in S}\sum_{k\in K\setminus K'}\sum_{p\in P_i} z_{ip}^k$, i.e., the number of nodes in set S minus the number of nodes in S that are served by a vehicle in set K' but not on any day in T' and the number of nodes in S that are not asigned to vehicles in set K'. Hence, in a feasible solution, we must have

$$\sum_{t\in T'}\sum_{k\in K'} x^{tk}(\delta(S)) \geq 2\left\lceil\frac{|S| - \sum_{i\in S}\sum_{k\in K'}\sum_{p\in P_i:p\cap T'=\emptyset} z_{ip}^k - \sum_{i\in S}\sum_{k\in K\setminus K'}\sum_{p\in P_i} z_{ip}^k}{q}\right\rceil.$$

Clearly, the right-hand side of this inequality is greater than or equal to the right-hand side of the inequality (13). □

Proposition 3 *Let $T' \subseteq T$ and $S \subseteq V \setminus \{0\}$. The inequality*

$$\sum_{t\in T'}\sum_{k\in K} x^{tk}(\delta(S)) \geq 2\left\lceil\frac{\sum_{i\in S}\eta_i(T')}{q}\right\rceil \tag{14}$$

where $\eta_i(T') = \min_{p\in P_i} |T'\cap p|$ for $i \in S$ is a valid inequality for set X.

Proof. As $\sum_{t\in T'}\sum_{k\in K} y_i^{tk} \geq \eta_i(T')$ for each $i \in S$ and

$$\sum_{t\in T'}\sum_{k\in K} x^{tk}(\delta(S)) \geq 2\left\lceil\frac{\sum_{i\in S}\sum_{t\in T'}\sum_{k\in K} y_i^{tk}}{q}\right\rceil$$

is satisfied by every feasible solution, the inequality (14) is valid. □

4 Branch-and-cut algorithm

We have devised a branch-and-cut algorithm to optimally solve the PVRP-DC. An algorithm of this type combines a branch-and-bound method for exploring a decision tree, and a cutting-plane method that computes lower bounds by solving LP relaxations improved by valid inequalities. We describe next the main features of our algorithm.

4.1 Preprocessing

Before starting the branch-and-bound search we perform a preprocessing phase in order to reduce the problem size and complexity.

4.1.1 Variable fixing

Some variables can be fixed to zero as follow. For each customer $i \in V \setminus \{0\}$ and each period $t \in T$ we check whether t belongs to any $p \in P_i$ or not. If the answer is negative, that is, if t does not appear in any of the allowed visits schedules for customer i, then we set $y_i^{tk} = 0$ for all vehicles $k \in K$.

4.1.2 Symmetry breaking

The PVRP-DC has some inherent symmetries in its definition, and counting with rules to avoid them is necessary to solve it efficiently We apply the following symmetry breaking strategy. For each vehicle $k \in \{1, ..., |K| - 1\}$, we choose a customer $i_k \in V \setminus \{0\}$ and set $\sum_{p \in P_{i_k}} z_{i_k,p}^{k'} = 0$ for all vehicles $k' \in K$ such that $k' > k$. This prevents i_k to be assigned to any vehicle with index larger than k. We choose i_k as the still non-assigned customer with largest visit frequency.

4.2 Initialization

The first LP model solved at the root node of the branch-and-cut algorithm is the linear relaxation of the model (1)–(9), excluding the connectivity constraints (5), which are exponential in number.

4.3 Cutting plane phase

Given a fractional solution (x^*, y^*, z^*) the separation routines for inequalities (10), (11), (5), (12), (13), and (14) are applied, in this sequence, and only if no violated cuts of previous families have been found (with the exception of the separation routine for (13), that is applied if no violated cuts (12) have been found). Moreover, the violation of inequalities (12), (13), and (14) is checked only at the root node, and the number of cuts of each family added to the model at each cutting plane iteration is limited to 100. These restrictions are set in order to improve the general performance of the algorithm, both in terms of consumed memory and time.

Constraints (11) are separated exactly by simple enumeration. We next outline the separation procedures for the other families of inequalities.

4.3.1 Separation of inequalities (5)

The inequalities (5) involving set $S = V \setminus \{0\}$ reduce to (10), and they can be separated by complete enumeration. To separate the general connectivity constraints (5) we use an exact polynomial procedure similar to the one used in Labbé et al. [28], inspired in the known separation algorithm of the subtour elimination constraints for the travelling salesman problem. The procedure consists of solving max-fow/min-cut problems on appropriately defined support graphs. To this end, for each given $i \in V \setminus \{0\}$, $k \in K$, and $t \in T$, we define a support graph $G' = (V', E')$ with $V' = V$ and $E' = \{e \in E : x_e^{*tk} > 0\}$. The capacity of all edges $e \in E'$ is fixed to x_e^{*tk}. Then we find a min-cut set $S \subset V'$ with $i \in S$ and $0 \notin S$, and we check the violation of inequality (5) for that set.

4.3.2 Separation of inequalities (12)

Inequalities (12) are separated exactly with a procedure that is also inspired in the classical separation of the subtour elimination constraints for the TSP. For $S \subseteq V \setminus \{0\}$, $k \in K$, and $t \in T$ we can rewrite (12) as:

$$qx^{tk}(E(0:S)) + (q-2) \sum_{i \in V \setminus \{S \cup \{0\}\}} x^{tk}(E(i:S)) + 2\sum_{i \notin S} y_i^{tk} \geq 2\sum_{i \in V} y_i^{tk}.$$

For each given $k \in K$ and $t \in T$, let us consider a support graph $G' = (V', E')$ with $V' = V \cup \{s\}$ where s is a dummy node and edge set E' is defined as follows:

- All edges $e \in E$ such that $x_e^{*tk} > 0$, each one with capacity qx_e^{*tk} if $e \in \delta(0)$ and capacity $(q-2)x_e^{*tk}$ otherwise.
- All edges connecting s with nodes $i \in V \setminus \{0\}$, each one with capacity $2y_i^{*tk}$.

A set $S \subset V'$ with $s \in S$ and $0 \notin S$ defines a violated inequality (12) if the capacity of the cut $\delta(S)$ on G' is smaller than $2\sum_{i \in V} y_i^{*tk}$. Hence, inequalities (12) are separated exactly by solving a min cut problem for each $k \in K$ and $t \in T$.

4.3.3 Separation of inequalities (13)

Inequalities (13) are defined for each $T' \subseteq T$, $K' \subseteq K$ and $S \subseteq V \setminus \{0\}$, and they can be written as

$$\sum_{t\in T'}\sum_{k\in K'} x^{tk}(\delta(S)) - 2\left\lceil \frac{|S|}{q} \right\rceil + 2\sum_{i\in S}\sum_{k\in K'}\sum_{p\in P_i : p\cap T'=\emptyset} z^k_{ip} + 2\sum_{i\in S}\sum_{k\in K\setminus K'}\sum_{p\in P_i} z^k_{ip} \geq 0.$$

As it seems complicated to optimally choose the S , T', and K' that minimize the left hand side of this inequality, we propose a heuristic method to separate (13). The method is based on the following observation: for a given solution (x^*, y^*, z^*), the left hand side of the inequality is

$$\sum_{t\in T'}\sum_{k\in K'} x^{*tk}(\delta(S)) - 2\left\lceil \frac{|S|}{q} \right\rceil + 2\sum_{i\in S}\sum_{k\in K'}\sum_{p\in P_i : p\cap T'=\emptyset} z^{*k}_{ip} + 2\sum_{i\in S}(1 - \sum_{k\in K'}\sum_{p\in P_i} z^{*k}_{ip}),$$

which is the same as

$$2\left(|S| - \left\lceil \frac{|S|}{q} \right\rceil\right) + \sum_{k\in K'}\left(\sum_{t\in T'} x^{*tk}(\delta(S)) + 2\sum_{i\in S}\sum_{p\in P_i : p\cap T'=\emptyset} z^{*k}_{ip} - 2\sum_{i\in S}\sum_{p\in P_i} z^{*k}_{ip}\right).$$

Hence, for some given S and T', a minimizing subset of vehicles is

$$K(S,T') = \{k \in K : \sum_{t\in T'} x^{*tk}(\delta(S)) + 2\sum_{i\in S}\sum_{p\in P_i : p\cap T'=\emptyset} z^{*k}_{ip} - 2\sum_{i\in S}\sum_{p\in P_i} z^{*k}_{ip} < 0\}.$$

Using this result, we separate inequalities (13) heuristically as follows. We consider only subsets T' that are singletons (i.e., that have cardinality one) and subsets S that violate a constraint (5). For each combination of such T' and S, we look for the subset $K' = K(S,T')$ as defined above, and we check the violation of inequality (13).

4.3.4 Separation of inequalities (14)

We use the following heuristic procedure to separate inequalities (14). First, we generate all the subsets $T' \subseteq T$ with cardinality 1, 2 and 3. Then, for each of those sets T' we look for a subset $S \subseteq V \setminus \{0\}$ that violates

$$\sum_{t\in T'}\sum_{k\in K} x^{tk}(\delta(S)) \geq 2\frac{\sum_{i\in S}\eta_i(T')}{q}, \tag{15}$$

which is equivalent, since $\sum_{i\in S}\eta_i(T') = \sum_{i\in V\setminus\{0\}}\eta_i(T') - \sum_{i\in V\setminus(S\cup\{0\})}\eta_i(T')$, to

$$\sum_{t\in T'}\sum_{k\in K} x^{tk}(\delta(S)) + 2\frac{\sum_{i\in V\setminus(S\cup\{0\})}\eta_i(T')}{q} \geq 2\frac{\sum_{i\in V\setminus\{0\}}\eta_i(T')}{q}.$$

To this end, we create a support graph $G' = (V', E')$ with $V' = V \cup \{s\}$, s being a dummy node, and edge set E' composed of all edges $e \in E$, with capacity $\sum_{t\in T'}\sum_{k\in K} x^{*tk}_e$, and all edges $\{i, s\}$, for all $i \in V \setminus \{0\}$, with capacity $2\eta_i(T')/q$. We find a min-cut S' in G' such that $s \in S'$ and $0 \notin S'$. We let $S = S' \setminus \{s\}$ and check whether sets S and T' give a violated inequality (14).

5 Computational results

The branch-and-cut algorithm was implemented in C++ and run on a personal computer with an Intel Core i7 CPU at 3.4 GHz and 16 GB of RAM. We used CPLEX 12.5 as mixed integer linear programming solver. Default settings for CPLEX were used, except for the variable selection strategy that was set to "strong branching". To solve the min-cut problems we used the routine included in the *Concorde TSP* software package.

5.1 Test instances

Our first intention was to evaluate the algorithm on the standard benchmark PVRP instances from the literature, the so called "old data". This is a group of 32 instances proposed by Christofides and Beasley [12], Russel and Igo [37], Russel and Gribbin [36], and Chao et al. [11], and used by many authors like Baldacci et al. [3]. However, we found that these instances are highly symmetric both in terms of the spatial distribution of the nodes and in terms of the allowed visit schedules of the customers. This results in PVRP solutions that are driver consistent, even if this feature is not required. In other words, the PVRP and PVRP-DC solutions coincide on these instances. As an example, Figure 3 shows an optimal PVRP solution for the instance `p14`, a case with 20 customers, two vehicles, and a planning horizon of four periods. The optimal solution has a cost equal to 954.81 and the two vehicles are used each period. It is clear from the figure that the solution is driver consistent, since customers to the left of the depot can always be visited by one of the vehicles, and customers to the right can be visited by the other one. We show the routes of the two vehicles in dashed and solid lines.

Therefore we decided to generate our own benchmark instances, aiming to produce test cases where driver consistency would not be implicit in the solution of the PVRP. To this end, we generated instances with a number of nodes n in $\{11, 21, 31, 41, 51, 61, 71\}$. Node coordinates are randomly generated in $[0, 100] \times [0, 100]$. The depot is placed at node 0 and the customers at the other nodes. Edge costs c_{ij} are computed as the Euclidean distance between i and j. The number of periods τ varies between 2 and 5, and m, the number of vehicles available at the depot, varies between 2 and 4 (except when $n = 11$, that takes values 2 and 3). Vehicle capacity is set to $q = \lceil 0.75(|V| - 1)/m \rceil$. For each node we generate randomly a visit frequency between 1 and τ, and a random number of allowed visit schedules with that frequency. We generated three instances for each combination of n, τ and m, resulting in a test bed with 240 instances. The whole set is available from the authors upon request.

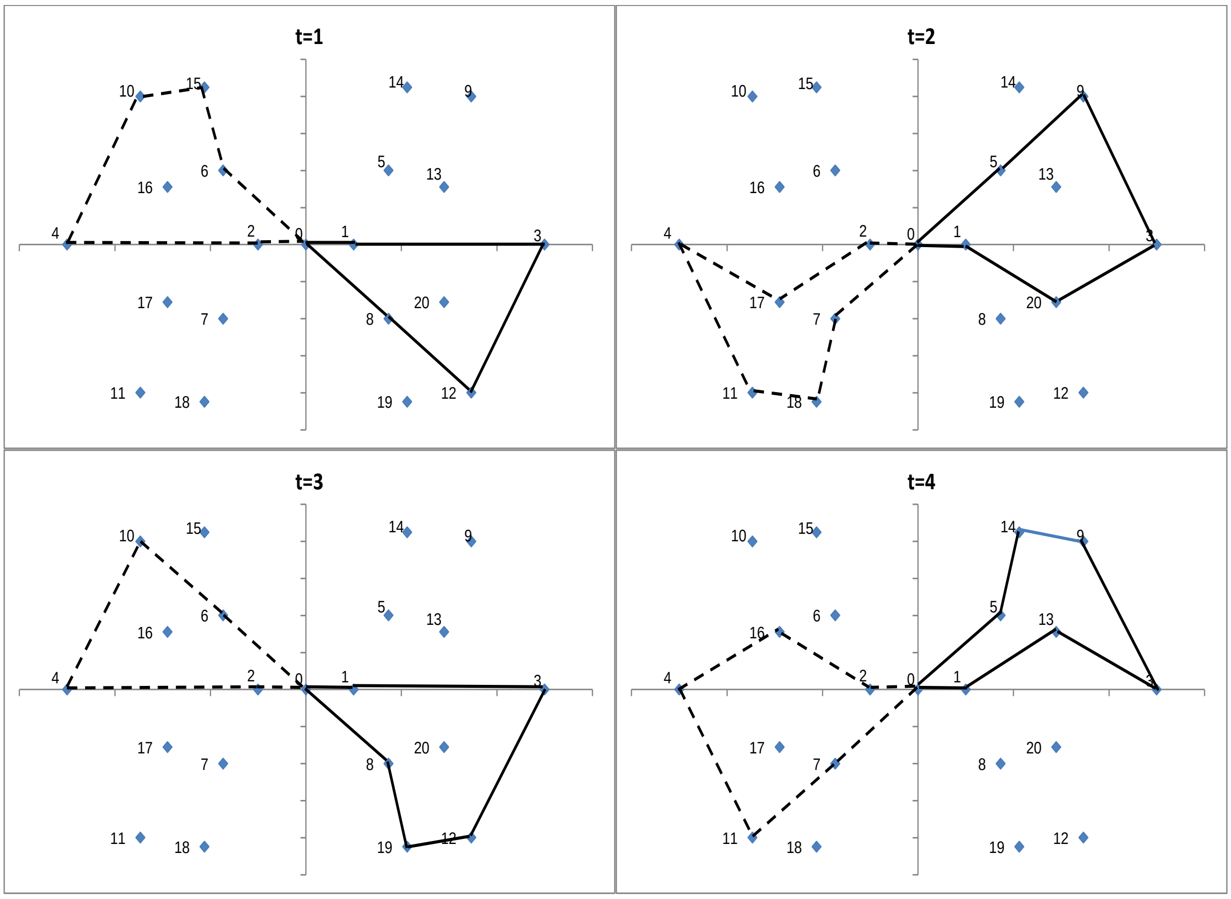

Figure 3: PVRP optimal solution for instance `p14` from the literature (cost = 954.81)

5.2 Evaluation of the algorithm

We ran the branch-and-cut algorithm on each instance with a time limit of two hours. Tables 1 and 2 show the results obtained for the small and medium sized instances, and for the large instances, respectively. Each line of the tables reports results corresponding to the three instances with those number of nodes n, number of periods τ, number of vehicles m, and vehicle capacity q. The displayed data are:

- *nOpt*: Number of instances solved to optimality within the time limit.
- *nFeas*: Number of instances for which only a feasible solution is available when the time limit is reached.
- *nInfeas*: Number of instances proved to be infeasible.
- *nUnk*: Number of instances for which there is neither a feasible solution, nor a proof of infeasibility, when the time limit is reached.

- *cpu*: Average computing time, in seconds.
- *BBnodes*: Average number of nodes in the search tree.
- *%-gap*: Average percentage gap between the optimal solution value and the lower bound at the end of the root node.
- *%-fgap*: Average percentage gap between the objective function value of the best solution found and the lower bound at the end of the computation.
- *nCuts*: Average number of generated cuts.

These tables show that, for a given number of nodes, the problem gets harder when the number of periods in the time horizon increases. Also, among all the instances with the same number of nodes and the same time horizon, the most computationally costly are those with 4 vehicles. We can see in Table 1 that all feasible instances with 11, 21 and 31 nodes are solved to optimality within the time limit. When the number of nodes is 41, the algorithm fails to find the optimal solution in six cases. They are two instances with 4 periods and 4 vehicles, and four instances with 5 periods and 3 or 4 vehicles. Table 2 shows the results for the largest instances, with 51, 61, and 71 nodes. The easiest instances with these sizes are those with 2 periods, though not all of them are solved to optimality when $m = 4$. For the largest time horizon ($\tau = 5$), only some instances with two vehicles are solved to optimality, and there are even some cases, with 4 vehicles (or even 3 vehicles and $n = 71$), for which the computation ends after two hours without finding even a feasible solution.

In order to evaluate the effect of the preprocessing and of the valid inequalities proposed in this work, we performed an experiment consisting in comparing the branch-and-cut algorithm with other five simplified versions of it. Table 3 shows the results obtained on the instances with 21 nodes, which we consider to be rather illustrative. The algorithms compared are:

- *B&C0*: It just solves the model (1)-(9). Constraints (5) are dynamically incorporated when violated.
- *B&C1*: It incorporates the preprocessing.
- *B&C2*: It includes also the separation of valid inequalities (11).
- *B&C3*: Valid inequalities (13) are also separated.
- *B&C4*: It incorporates the separation of inequalities (12).
- *complete B&C*: Complete branch-and-cut algorithm, as described in Section 4.

n	τ	m	q	nOpt	nFeas	nInfeas	nUnk	cpu	BBnodes	%-gap	%-fgap	nCuts
11	2	2	4	3	0	0	0	0.10	1.00	0.71	0.00	29.00
		3	3	3	0	0	0	0.20	0.00	0.00	0.00	56.00
	3	2	4	3	0	0	0	0.10	0.00	0.00	0.00	31.00
		3	3	3	0	0	0	0.36	10.67	5.50	0.00	133.67
	4	2	4	3	0	0	0	0.31	7.00	3.88	0.00	108.33
		3	3	3	0	0	0	0.74	19.67	5.83	0.00	183.33
	5	2	4	3	0	0	0	0.92	32.67	8.67	0.00	228.33
		3	3	3	0	0	0	13.23	413.00	10.96	0.00	529.00
21	2	2	8	3	0	0	0	0.62	24.67	2.42	0.00	276.00
		3	5	1	0	2	0	0.80	6.00	1.45	0.00	401.00
		4	4	3	0	0	0	11.98	159.00	4.64	0.00	890.00
	3	2	8	3	0	0	0	0.74	4.33	1.85	0.00	344.67
		3	5	3	0	0	0	8.24	103.67	5.57	0.00	880.00
		4	4	3	0	0	0	49.27	581.00	7.35	0.00	1414.00
	4	2	8	3	0	0	0	2.01	20.00	4.42	0.00	534.67
		3	5	3	0	0	0	9.93	77.67	7.03	0.00	1061.00
		4	4	3	0	0	0	555.12	3363.67	9.90	0.00	2957.00
	5	2	8	3	0	0	0	9.14	80.00	7.06	0.00	1094.33
		3	5	3	0	0	0	58.48	189.00	7.89	0.00	2031.67
		4	4	3	0	0	0	149.82	904.67	7.20	0.00	2961.67
31	2	2	12	3	0	0	0	1.77	4.67	1.48	0.00	447.00
		3	8	3	0	0	0	14.62	81.00	3.87	0.00	1337.00
		4	6	3	0	0	0	51.14	227.00	4.40	0.00	2046.33
	3	2	12	3	0	0	0	3.31	7.33	0.59	0.00	641.67
		3	8	3	0	0	0	22.27	93.33	5.97	0.00	1942.67
		4	6	3	0	0	0	415.99	1975.00	7.43	0.00	3497.67
	4	2	12	3	0	0	0	14.83	12.00	2.80	0.00	1284.67
		3	8	3	0	0	0	64.39	201.67	7.82	0.00	2982.00
		4	6	3	0	0	0	372.66	1011.67	6.77	0.00	5117.00
	5	2	12	3	0	0	0	9.44	20.00	5.38	0.00	1571.33
		3	8	3	0	0	0	345.93	1341.00	8.01	0.00	4471.33
		4	6	3	0	0	0	1826.53	3622.67	8.43	0.00	7059.33
41	2	2	15	2	0	1	0	12.36	16.50	1.50	0.00	1158.50
		3	10	2	0	1	0	81.87	137.50	4.20	0.00	3111.00
		4	8	3	0	0	0	191.31	269.00	3.72	0.00	4420.33
	3	2	15	3	0	0	0	27.28	25.00	2.70	0.00	1731.00
		3	10	3	0	0	0	78.13	63.67	2.97	0.00	3016.33
		4	8	3	0	0	0	3092.43	4629.67	6.37	0.00	9505.67
	4	2	15	3	0	0	0	161.06	139.67	4.82	0.00	5119.33
		3	10	3	0	0	0	1908.42	1910.00	8.09	0.00	10445.67
		4	8	1	2	0	0	4890.36	2106.33	6.62	2.45	13868.33
	5	2	15	3	0	0	0	558.67	517.67	5.53	0.00	6727.33
		3	10	2	1	0	0	4050.18	1757.00	9.66	2.30	15980.00
		4	8	0	3	0	0	7200.00	1929.00	8.38	3.10	20456.00

Table 1: Branch-and-cut results for small and medium-sized instances

n	τ	m	q	nOpt	nFeas	nInfeas	nUnk	cpu	BBnodes	%-gap	%-fgap	nCuts
51	2	2	19	3	0	0	0	27.79	20.00	1.55	0.00	1862.67
		3	13	3	0	0	0	87.10	91.00	2.35	0.00	3522.67
		4	10	3	0	0	0	1273.16	1359.00	4.47	0.00	8900.33
	3	2	19	3	0	0	0	181.67	254.33	3.00	0.00	5137.00
		3	13	3	0	0	0	260.37	144.00	2.97	0.00	5907.67
		4	10	0	3	0	0	7200.00	1747.33	6.64	3.90	19496.67
	4	2	19	3	0	0	0	838.16	389.00	5.50	0.00	10448.33
		3	13	0	3	0	0	7200.00	1438.67	9.54	4.94	22766.67
		4	10	1	2	0	0	6436.24	944.67	8.29	5.68	23622.67
	5	2	19	3	0	0	0	1347.64	532.67	5.07	0.00	12951.00
		3	13	0	3	0	0	7200.00	851.67	9.07	5.25	23697.67
		4	10	0	3	0	0	7200.00	580.00	8.95	7.36	24142.00
61	2	2	23	3	0	0	0	99.57	87.67	2.53	0.00	4738.33
		3	15	2	0	1	0	1694.72	2246.50	2.17	0.00	8288.00
		4	12	2	1	0	0	4043.11	1238.33	3.63	0.07	16114.67
	3	2	23	3	0	0	0	515.46	265.00	2.94	0.00	8565.33
		3	15	3	0	0	0	4242.93	911.67	3.48	0.00	15412.67
		4	12	0	3	0	0	7200.00	606.00	7.37	6.25	22824.00
	4	2	23	3	0	0	0	917.42	330.33	3.76	0.00	10765.33
		3	15	1	2	0	0	7126.29	950.33	6.58	2.06	26000.67
		4	12	0	3	0	0	7200.00	388.33	8.65	7.23	26165.33
	5	2	23	2	1	0	0	2689.71	723.33	4.70	0.37	15068.67
		3	15	0	3	0	0	7200.00	611.67	8.01	5.06	28286.33
		4	12	0	2	0	1	7200.00	463.50	9.13	7.79	28849.50
71	2	2	27	3	0	0	0	187.42	309.33	2.63	0.00	7221.33
		3	18	3	0	0	0	2177.94	680.00	3.61	0.00	14567.67
		4	14	2	1	0	0	5413.79	1270.00	4.32	1.44	20967.33
	3	2	27	3	0	0	0	464.24	84.67	1.96	0.00	7226.33
		3	18	3	0	0	0	3263.23	474.00	4.28	0.00	18308.00
		4	14	0	3	0	0	7200.00	473.00	7.80	6.58	23040.33
	4	2	27	2	1	0	0	3054.23	494.00	3.84	0.68	21470.33
		3	18	0	3	0	0	7200.00	375.00	10.41	8.51	25649.33
		4	14	0	2	0	1	7200.00	347.50	11.44	10.91	24336.00
	5	2	27	1	2	0	0	6579.80	1033.00	4.13	0.20	28378.67
		3	18	0	1	0	2	7200.00	331.00	8.56	8.23	31394.00
		4	14	0	1	0	2	7200.00	100.00	9.09	8.85	22796.00

Table 2: Branch-and-cut results for large instances

				B&C0		B&C1		B&C2		B&C3		B&C4		complete B&C	
n	τ	m	q	cpu	%-gap	cpu	%-gap	cpu	%-gap	cpu	%-gap	cpu	%-gap	cpu	%-gap
21	2	2	8	0.66	8.13	0.31	3.49	0.27	3.49	0.23	3.31	0.20	3.26	0.33	1.44
		3	5	5.29	11.51	0.72	7.28	1.08	7.28	0.69	5.70	1.70	4.38	0.80	1.45
		4	4	35.79	11.60	2.67	7.93	1.76	7.90	1.86	5.07	5.12	2.80	2.14	1.68
	3	2	8	1.20	8.59	0.59	7.34	0.50	7.34	0.55	7.34	1.47	5.39	0.72	3.90
		3	5	29.72	15.64	10.95	14.80	6.69	14.80	7.16	11.16	21.31	8.14	15.87	6.28
		4	4	t.l.	19.37	111.03	17.52	89.93	17.52	53.04	13.80	177.65	10.62	124.15	9.92
	4	2	8	3.87	10.51	1.20	5.78	2.34	5.78	1.79	5.78	3.67	4.75	2.04	3.42
		3	5	240.05	14.34	9.28	10.68	7.18	10.68	4.87	10.45	17.67	6.64	10.72	6.61
		4	4	t.l.	15.18	84.94	13.23	39.75	13.23	41.36	11.62	69.16	7.66	97.22	7.34
	5	2	8	28.81	12.10	17.50	9.13	9.77	9.11	5.48	9.10	17.22	7.25	15.46	7.24
		3	5	6599.03	16.90	170.43	15.31	223.22	15.28	143.26	12.30	203.08	8.27	142.82	7.74
		4	4	t.l.	15.61	577.55	13.58	187.17	13.52	607.23	10.37	360.30	6.57	318.80	6.20

Table 3: Effect of using preprocessing and valid inequalities

In each of these methods, the separation procedures for the different types of valid inequalities are applied in the same sequence that is used in the complete branch-and-cut. Each line of the table shows average computing times and gaps at the end of the root node for the three instances with those values of n, τ, m and q. The term "t.l." in the *cpu* column indicates that the two hours time limit was reached.

Note that the most basic algorithm (*B&C0*) fails to solve all the instances with four vehicles and a time horizon of more than 2 days within the fixed time limit. The use of the preprocessing produces already a great improvement of the results (see columns under *B&C1*). From that point, the introduction of each family of valid inequalities considered helps to gradually reduce the gaps. Regarding the computing times, the improvement is not constant, since they increase in some cases when a new family of valid inequalities is separated. However, the complete branch-and-cut algorithm gives the best results for the hardest instances, those with a time horizon of five days.

Table 4 shows the average percentage of violated cuts of the different families of valid inequalities presented in Section 3. Each line reports the results corresponding to the instances with given size n (i.e., for 24 instances when $n = 11$, and 36 instances when $n \geq 21$). The largest number of violated cuts generated corresponds to the valid inequalities (11). In fact these inequalities, that are separated at the beginning of the cutting plane phase and in all the nodes of the search tree, sum up to around 60% of the added cuts (almost 88% for instances with $n = 11$). The second place is for the inequalities (12), which are separated exactly. The

n	(11)	(12)	(13)	(14)
11	87.98	1.17	2.50	8.35
21	52.18	28.97	10.28	8.06
31	64.47	24.05	3.65	7.82
41	61.70	21.67	3.68	12.94
51	57.21	20.59	7.30	14.90
61	62.41	17.10	6.13	14.36
71	58.48	18.29	9.52	13.71

Table 4: Average percentage of violated valid inequalities

last two positions correspond, in general, to inequalities (14) and (13), respectively. The last three families of inequalities are separated only in the root node, for the sake of efficiency. For this reason their number is much smaller than that of inequalities (11). Moreover, the separation procedures for (13) and (14) are heuristic, and therefore they might fail to detect existing violated cuts.

Finally, we conducted a computational experiment to evaluate the effect of the number of visit schedules in the problem complexity. We modified our instances so that there would be only one visit pattern associated to each customer, and we solved them. As expected the modified instances are easier to handle, and on average they are solved in 75% less computing time.

5.3 The cost of consistency

In this section we try to analyze the effect of the consistency requirement. To this end, we show in Table 5 the optimal PVPR and PVRP-DC costs for the 24 instances from our benchmark with 11 nodes. Column *#-incosis* reports the number of customers that are visited by different drivers along the time horizon in the PVRP solution, and column *%-increase* shows the percentage increment in the solution cost when driver consistency is required. The average percentage increment and average number of inconsistencies are given in the last line.

This experiment shows that the optimal solutions of the PVRP and PVRP-DC are different in all but 3 of the 24 cases considered. That is, only in 3 instances out the 24 the PVRP solution resulted to be driver consistent without having required it. On the contrary, in most cases the PVRP solution includes several customers that are visited by different drivers in different periods. To provide a consistent service in those cases, a cost increase must be incurred. The percentage cost increment goes form 0.79% to 10.65%, and it is 3.99% on average.

Based on these results, we can conclude that driver consistency is not a natural characteristic of the PVRP in general instances and that imposing it modifies the problem.

n	τ	m	q	instance	PVRP-cost	PVRP-DC-cost	%-increase	#-inconsis
11	2	2	4	a	688.64	697.95	1.33	5
				b	633.14	660.95	4.21	1
				c	657.03	657.03	0.00	0
		3	3	a	783.22	790.14	0.88	6
				b	701.30	701.30	0.00	0
				c	732.10	744.25	1.63	3
	3	2	4	a	924.16	936.60	1.33	5
				b	1032.32	1032.32	0.00	0
				c	752.53	759.63	0.93	1
		3	3	a	1017.67	1123.86	9.45	7
				b	1073.92	1189.47	9.71	7
				c	825.68	832.27	0.79	5
	4	2	4	a	1214.78	1276.69	4.85	7
				b	1334.26	1400.05	4.70	7
				c	1077.32	1104.27	2.44	5
		3	3	a	1353.87	1457.34	7.10	5
				b	1443.44	1502.87	3.95	8
				c	1156.43	1203.33	3.90	5
	5	2	4	a	1186.08	1242.86	4.57	1
				b	1605.76	1759.93	8.76	7
				c	1224.82	1248.36	1.89	8
		3	3	a	1353.07	1432.09	5.52	7
				b	1688.87	1894.51	10.85	9
				c	1317.23	1414.96	6.91	6
average							3.99	5.29

Table 5: Effect of consistency

6 Conclusions

In this paper we have addressed a complex routing problem in which a fleet of homogeneous capacitated vehicles has to give service to a number of customers over a planning horizon of several periods. Moreover, each customer has to be visited according to one of its possible visit schedules, and always by the same vehicle/driver. Solving the problem implies to choose a visit schedule for each customer, and to design the vehicles' routes for each period of the time horizon respecting the driver consistency requirements and the capacity restrictions of the vehicles.

We present, for this new variant of the Periodic VRP, a mathematical model and several families of valid inequalities. We describe an exact branch-and-cut algorithm, and show computational results on instances with up to 71 nodes and different time horizons and number of vehicles. The proposed algorithm is able to solve to optimality most of the instances in a reasonable amount of time.

In the applications where driver consistency is required, the drivers give a service at the customer nodes they visit. Certainly, the service time has a high impact on the quality and/or the utility of this service. An interesting future extension of the current study is to decide on the time that the drivers spend at each customer node in order to maximize the quality/utility of their service.

The PVRP-DC could also be extended to consider multi-trips, that is, to allow that a single vehicle/driver performs several consecutive routes at each period. In this case, additional constraints on the number of trips by a driver, or on the total number of customers that a driver can serve, have to be imposed so that the driver consistency requirement still makes sense.

Finally, as it is clear that solving to optimality the PVRP-DC is a difficult task even on medium-sized instances, another interesting field for future research is the design of efficient heuristic algorithms able to provide good quality solutions in short computing times.

Acknowledgements

This work has been partially supported by the research projects ProID2017010132 (Gobierno de Canarias) and MTM2015-63680-R (MINECO/FEDER).

References

[1] Alegre. J., Laguna, M., and Pacheco, J. (2007). Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts. *European Journal of Operational Research*, 179,

736–746.

[2] An, Y.-J., Kim, Y.-D., Jeong, B.J., and Kim, S.-D. (2012). Scheduling healthcare services in a home healthcare system. *Journal of the Operational Research Society*, 63, 1589–1599.

[3] Baldacci, R., Bartolini, E., Mingozzi, A., and Valletta, A.(2011). An Exact Algorithm for the Period Routing Problem. *Operations Research*, 59 (1), 228–241.

[4] Baptista, S., Oliveira, R.C., and Zúquete, E. (2002). A period vehicle routing case study. *European Journal of Operational Research*, 139 (2), 220–229.

[5] Banerjea-Brodeur, M., Cordeau, J.-F., Laporte, G., and Lasry, A.(1998). Scheduling linen deliveries in a large hospital. *Journal of the Operational Research Society*, 49 (8), 777–780.

[6] Beltrami, E.J., and Bodin, L.D. (1974). Networks and vehicle routing for municipal waste collection. *Networks* 4 (1), 65–94.

[7] Blakely, F., Bozkaya, B., Cao, B., Hall, W., and Knolmajer, J. (2003). Optimizing periodic maintenance operations for Schindler Elevator Corporation. *Interfaces*, 33 (1), 67–79.

[8] Bommisetty, D., Dessouky, M., and Jacobs, L. (1998). Scheduling collection of recyclable material at Northern Illinois University campus using a two-phase algorithm. *Computers and Industrial Engineering*, 35 (3–4), 435–438.

[9] Butler, M., Williams, H.P., and Yarrow, L.-A. (1997). The two-period travelling salesman problem applied to milk collection in Ireland. *Computational Optimization and Applications*, 7 (3), 291–306.

[10] Campbell, A.M., and Wilson, J.H. (2014). Forty years of periodic vehicle routing. *Networks*, 63 (1), 2–15.

[11] Chao, I.-M., Golden, B.L., and Wasil, E.A. (1995). An improved heuristic for the period vehicle routing problem. *Networks*, 26 (1), 25–44.

[12] Christofides, N., and Beasley, J.E. (1984). The period routing problem. *Networks*, 14 (2), 237–256.

[13] Claassen, G.D.H., and Hendriks, T.H.B. (2007). An application of Special Ordered Sets to a periodic milk collection problem. *European Journal of Operational Research*, 180 (2), 754–769.

[14] Coene, S., Arnout, A., and Spieksma, F.C.R. (2010). On a periodic vehicle routing problem. *Journal of the Operational Research Society*, 61, 1719–1728.

[15] Cordeau, J.-F., Gendreau, M., and Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems. *Networks*, 30 (2), 105–119.

[16] Drummond, L.M.A., Ochi, L.S., and Vianna, D.S. (2001). An asynchronous parallel metaheuristic for the period vehicle routing problem. *Future Generation Computer Systems*, 17 (4), 379–386.

[17] Francis, P.M., Smilowitz, K.R., and Tzur, M. (2006). The period vehicle routing problem with service choice. *Transportation Science*, 40 (4), 439–454.

[18] Francis, P.M., Smilowitz, K.R., and M. Tzur, M. (2008). The period vehicle routing problem and its extensions. In *The vehicle routing problem: latest advances and new challenges.* Springer US, 73–102.

[19] Gaudioso, M., and Paletta, G. (1992). A heuristic for the period vehicle routing problem. *Transportation Science*, 26 (2), 86–92.

[20] Gaur, V., and Fisher, M.L. (2004). A periodic inventory routing problem at a supermarket chain. *Operations Research*, 52 (6), 813–822.

[21] Golden, B.L., and Wasil, E.A. (1987). Computerized vehicle routing in the soft drink industry. *Operations Research*, 35 (1), 6–17.

[22] Groër, C., Golden, B., and Wasil, E. (2009). The consistent vehicle routing problem. *Manufacturing & service operations management*, 11 (4), 630–643.

[23] Hadjiconstantinou, E., and Baldacci, R. (1998). A multi-depot period vehicle routing problem arising in the utilities sector. *Journal of the Operational Research Society*, 49, 1239–1248.

[24] Hemmelmayr, V.C., Doerner, K.F., Hartl, R.F., and Savelsbergh, M.W.P. (2009). Delivery strategies for blood products supplies. *OR Spectrum*, 31 (4), 707–725.

[25] Hemmelmayr, V.C., Doerner, K.F., and Hartl, R.F. (2009). A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195, 791–802.

[26] Jang, W., Lim, H.H., Crowe, T.J., Raskin, G., and Perkins, T.E. (2006). The Missouri Lottery optimizes its scheduling and routing to improve efficiency and balance. *Interfaces*, 36 (4), 302–313.

[27] Kovacs, A.A., Golden, B.L., Hartl, R.F., and Parragh, S.N. (2015) The generalized consistent vehicle routing problem. *Transportation Science*, 49 (4), 796–816.

[28] Labbé, M., Laporte, G., Rodríguez-Martín, I., and Salazar-González, J.J. (2004). The ring star problem: Polyhedral analysis and exact algorithm. *Networks*, 43 (3), 177–189.

[29] le Blanc, H.L., Cruijssen, F., Fleuren, H.A., and de Koster, M.B.M. (2006). Factory gate pricing: An analysis of the Dutch retail distribution. *European Journal of Operational Research*, 174 (3), 1950–1967.

[30] Letchford, A.N., Eglese, R.W., and Lysgaard, J. (2002). Multistars, partial multistars and the capacitated vehicle routing problem. *Mathematical Programming*, 94, 21–40.

[31] Luo, Z., Qin, H., Che, C.H., and Lim, A. (2015). On service consistency in multi-period vehicle routing. *European Journal of Operational Research*, 243 (3), 731–744.

[32] Maya, P., Sorensen, K., and Goos, P. (2012). A metaheuristic for a teaching assistant assignment-routing problem. *Computers & Operations Research*, 39, 249–258.

[33] Mourgaya, M., and Vanderbeck, F. (2007). Column generation based heuristic for tactical planning in multi-period vehicle routing. *European Journal of Operational Research*, 183 (3), 1028–1041.

[34] Nuortio, T., Kytöjoki, J., Niska, H., and Bräysy, O. (2006). Improved route planning and scheduling of waste collection and transport. *Expert Systems with Applications*, 30, 223–232.

[35] Ronen, D., and Goodhart, C.A. (2007). Tactical store delivery planning. *Journal of the Operational Research Society*, 59, 1047–1054.

[36] Russell, R.A., and Gribbin, D. (1991). A multiphase approach to the period routing problem. *Networks*, 21 (7), 747–765.

[37] Russell, R.A., and Igo, W. (1979). An assignment routing problem. *Networks*, 9 (1), 1–17.

[38] Shih, L.-H., and Chang, H.-C. (2001). A routing and scheduling system for infectious waste collection. *Environmental Modeling and Assessment*, 6(4), 261–269.

[39] Shih, L.-H., and Lin, Y.-T. (1999). Optimal routing for infectious waste collection. *Journal of Environmental Engineering*, 125 (5), 479–484.

[40] Tan, C.C.R., and Beasley, J.E. (1984). A heuristic algorithm for the period vehicle routing problem. *Omega*, 12(5), 497–504.

[41] Teixeira, J., Antunes, A.P., and de Sousa, J.P. (2004). Recyclable waste collection planning - a case study. *European Journal of Operational Research*, 158 (3), 543–554.

[42] Zhu, J., Zhu, W., Che, C.H., and Lim, A. (2008) A vehicle routing system to solve a periodic vehicle routing problem for a food chain in Hong Kong. In Proceedings of the 20th National Conference on Innovative Applications of Artificial Intelligence IAAI'08, Vol. 3, 1763–1768.