European Journal of Operational Research 291 (2021) 32–51

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

On the Distance-Constrained Close Enough Arc Routing Problem

Ángel Corberán[a], Isaac Plana[b], Miguel Reula[a], José M. Sanchis[c,*]

[a] Departament d'Estadística i Investigació Operativa, Universitat de València, Avda. Dr. Moliner 50, Burjassot 46100, Valencia, Spain
[b] Departament de Matemáticas para la Economía y la Empresa, Universitat de València, Avda. Tarongers s/n, Valencia 46022, Valencia, Spain
[c] Departament de Matemática Aplicada, Universidad Politécnica de Valencia, Camino de Vera s/n, Valencia 46022, Valencia, Spain

ARTICLE INFO

Article history:
Received 15 January 2020
Accepted 9 September 2020
Available online 16 September 2020

Keywords:
Routing
Distance constraints
Close-enough
Rural Postman
Branch and cut

ABSTRACT

Arc routing problems consist basically of finding one or several routes traversing a given set of arcs and/or edges that must be serviced. The Close-Enough Arc Routing Problem, or Generalized Directed Rural Postman Problem, does not assume that customers are located at specific arcs, but can be serviced by traversing any arc of a given subset. Real-life applications include routing for meter reading, in which a vehicle equipped with a receiver travels a street network. If the vehicle gets within a certain distance of a meter, the receiver collects its data. Therefore, only a few streets which are close enough to the meters need to be traversed. In this paper we study the generalization of this problem to the case in which a fleet of vehicles is available. This problem, the Distance-Constrained Close Enough Arc Routing Problem, consists of finding a set of routes with minimum total cost such that their length does not exceed a maximum distance.

In this article, we propose a new formulation for the Distance-Constrained Close Enough Arc Routing Problem and present some families of valid inequalities that we use in a branch-and-cut algorithm for its solution. Extensive computational experiments have been performed on a set of benchmark instances and the results are compared with those obtained with other heuristic and exact methods.

1. Introduction

Some real-world logistic problems, such as meter reading, waste collection or postal delivery, require that a service is performed while traversing a street or road. Recent technological advances allow some of these tasks to be performed in an easier and less expensive way. Particularly, radio frequency technology (RFID) permits collecting the consumption data from electricity, gas or water meters remotely (Uribe-Pérez, Hernández, De la Vega, & Angulo, 2016).

Until recently, the collection of this data had to be performed door to door and thus the vehicles or workers had to traverse all the streets where the meters where located. Using RFID, the service providers do not need to visit all their customers. The meter sends the data consumption and, if the receiver is closer than a certain distance, this data is collected. Therefore, the operator only needs to enter the meter covering zone to perform the service. An interesting summary of the models and methods proposed since the late 1970s in meter reading is the paper by Eglese, Golden, and Wasil (2014).

The first description of this application for a single vehicle was provided by Gulczynski, Heath, and Price (2006). They consider the problem where each customer is modeled as a point in the plane and the salesman must travel within a required radius r of each customer. They assume that the salesman "is not restricted to a road network", that is, it can move between any pair of points in the plane following a straight line whose cost is the Euclidean distance. The objective is to minimize the total distance traveled. Since then, this problem, the Close Enough Traveling Salesman Problem (CETSP), and variants where the radius associated with each customer may be different or the shape of the area around the customer is not a circle have been studied by several authors: Dong, Yang, and Chen (2007), Mennell (2009), Shuttleworth, Golden, Smith, and Wasil (2008), Behdani and Smith (2014), Coutinho, Subramanian, do Nascimento, and Pessoa (2016), and Carrabs, Cerrone, Cerulli, and Gaudioso (2017). Another closely related problem is the Covering Tour Problem (CTP) studied by Gendreau, Laporte, and Semet (1997). The CTP is defined on a graph $G = (V \cup W, E)$, where W is a set of vertices that must be covered, and consists of determining a minimum length Hamiltonian cycle on a subset of V such that every vertex of W is within a prespecified distance from the cycle. For this problem, the authors present an ILP formulation and several valid inequalities and propose a heuristic and a branch-and-cut algorithm.

* Corresponding author.
E-mail addresses: angel.corberan@uv.es (Á. Corberán), isaac.plana@uv.es (I. Plana), miguel.reula@uv.es (M. Reula), jmsanchis@mat.upv.es (J.M. Sanchis).

https://doi.org/10.1016/j.ejor.2020.09.012
0377-2217/© 2020 Elsevier B.V. All rights reserved.

Hà, Bostel, Langevin, and Rousseau (2012, 2014) consider the meter reading application in the context of a street network, where, although the customers (meters) do not need to be nodes of the network, they can be serviced by traversing a street that is close enough. Hà's et al. approach to the problem is clearly that of an arc routing problem. Unlike the previous articles, where the service has to be done in all or some of the vertices of a network (node or vehicle routing problems), in arc routing problems (ARPs) the service has to be done in some or all the arcs or/and edges of a network. See the book Corberán and Laporte (2014), the annotated bibliography by Mourão and Pinto (2017), and Corberán, Eglese, Hasle, Plana, and Sanchis (2020) for a comprehensive treatment of this area. Hà et al. call this problem the Close-Enough Arc Routing Problem (CEARP) and propose a formulation and a branch-and-cut algorithm that exhibits a very good performance on large instances. In a more general context, Drexl (2007, 2014) studies this problem and calls it the Generalized Directed Rural Postman Problem (GDRPP). In the GDRPP each customer has an associated subset of arcs, of which at least one has to be traversed in order to service the customer, and the goal is to find a minimum cost route servicing all the customers. Drexl proves that the problem is NP-hard because it contains the Directed Rural Postman Problem as a special case and proposes a formulation and a branch-and-cut algorithm producing good computational results.

Ávila, Corberán, Plana, and Sanchis (2016) introduce two new formulations for the CEARP, present a polyhedral study and propose a branch-and-cut algorithm using several families of new inequalities, comparing the obtained results with those from Hà, Bostel, Langevin, and Rousseau (2014). In Cerrone, Cerulli, Golden, and Pentangelo (2017) a new flow-based formulation is given, as well as some techniques to reduce the size of the graph. The results obtained on one of the set of instances proposed in Hà et al. (2012) using this new formulation improve those of Hà et al., but are slightly worse than those in Ávila et al. (2016). A stochastic version of the CEARP has been studied by Renaud, Absi, and Feillet (2017). In that paper, the authors point out that the remote reading of a meter may fail and therefore there is an uncertainty in the collection of the data. They introduce the probability of reading a meter as a function of the distance of the customer from the vehicle route, and propose a mathematical formulation and a cutting-plane algorithm and several heuristics for its solution.

The Generalized Arc Routing Problem is an undirected version of the CEARP where the clusters of edges associated with the customers are pairwise-disjoint connected subgraphs. This problem, which can be seen as the arc routing counterpart of the Generalized Traveling Salesman Problem, is studied by Aráoz, Fernández, and Franquesa (2017), who describe some facets and valid inequalities for the problem and present a branch-and-cut algorithm for its solution.

Other applications of the CETSP and the CEARP arise in the robot monitoring of wireless sensor networks (Behdani & Smith, 2014; Yuan, Orlowska, & Sadiq, 2007). As Yuan et al. (2007) point out, “in a wireless sensor network, where sensors are geographically distant from each other, it may not be practical to require sensors to directly coordinate with each other to form a communication network due to the energy restriction. One possible solution is to employ a mobile robot, which can travel to all sensors, to download the data and finally return to its base station (starting position)”. Like in meter reading, “the robot must be physically within its effective range”. Aráoz et al. (2017) also note that another area of application is in quality control for networks maintenance, where only a small subset of the edges of a network has to be traversed. The same authors argue that the CEARP is the most appropriate problem for modeling location/arc routing problems in which facilities have to be located at some given areas and connected among them by means of a route.

The CEARP is defined for a single vehicle, but in practical applications where the number of customers is very high the service must be carried out by a fleet of vehicles (or one vehicle performing several routes). Ávila, Corberán, Plana, and Sanchis (2017) introduce the problem of finding a set of routes with total minimum cost, that start and end at a depot, service all the customers, and such that the length of each route does not exceed a certain limit. For this problem, the Distance Constrained CEARP (DC-CEARP), the authors introduce four different formulations and, based on them, they propose four branch-and-cut algorithms for its solution. Recently, a matheuristic algorithm for the DC-CEARP has been described in Corberán, Plana, Reula, and Sanchis (2019).

In this paper we deepen the study of the DC-CEARP. The contribution of this work is threefold. First, we propose a new formulation for the DC-CEARP that combines the best features of the previously existing ones. For this formulation, an exhaustive study of its associated polyhedron is performed, and several different families of valid inequalities are proposed. Secondly, many of the new inequalities presented here can be used, directly or easily adapted, in other arc routing problems, and the ideas in which some of the algorithms designed for the separation of these inequalities are based (or the algorithms themselves), can be used for similar inequalities in other problems. Finally, the designed branch-and-cut algorithm is an efficient exact method that is able to solve instances with up to 140 customers, 196 vertices, 544 arcs, and 5 vehicles to optimality within two hours computing time.

The paper is organized as follows. In Section 2 we describe the problem formally and present the new formulation. Several families of valid inequalities are shown in Section 3, while the corresponding separation methods and the branch-and-cut algorithm are presented in Section 4. Computational experiments are reported in Section 5, and some conclusions and future lines of research are given in Section 6.

2. Problem definition and formulations

The Distance-Constrained Close Enough Arc Routing Problem, DC-CEARP, is defined as follows. Consider a strongly connected and directed graph $G = (V, A)$, where V is the set of vertices, A is the set of arcs, and, for each arc $(i, j) \in A$, there is a distance d_{ij} associated with its traversal. Vertex 1 represents the depot. There is a fleet of K identical vehicles based at the depot and a set of L customers. Each customer $c \in \{1, \dots, L\}$ has an associated set of arcs $H_c \subseteq A$ from which it can be serviced. We consider that a customer c is serviced if there is a vehicle k that traverses at least one arc in H_c. The length of the routes of the vehicles must not exceed a maximum travel distance denoted by D_{max}. The aim of the DC-CEARP is to find a set of K routes, starting and ending at the depot, with minimum total distance and such that each customer $c = 1, \dots, L$, is serviced and the length of each route does not exceed D_{max}.

In what follows, $\mathbb{K} = \{1, \dots, K\}$ will represent the set of vehicles, $\mathbb{H} = \{1, \dots, L\}$ the set of customers, and $A_R = H_1 \cup H_2 \cup \dots \cup H_L$ the set of required arcs. The arcs in the set $A_{NR} = A \setminus A_R$ are called non-required arcs. Given two sets $S, T \subset V$, we define $(S : T) = \{(i, j) \in A : i \in S,\ j \in T\}$ and $(S, T) = (S : T) \cup (T : S)$. In particular, $\delta^+(S) = (S : V \setminus S)$, $\delta^-(S) = (V \setminus S : S)$ and $\delta(S) = (S, V \setminus S)$. Finally, $A(S) = \{(i, j) \in A : i, j \in S\}$ and, given a set of variables x_{ij} indexed on the arcs, and given a set F of arcs, $x(F) = \sum_{(i,j) \in F} x_{ij}$.

In Ávila et al. (2017) four formulations for the DC-CEARP using different types of variables are presented. In these formulations, there are two types of variables. Some variables are associated with the number of times a vehicle traverses an arc, while other variables indicate if the vehicle traversing a required arc services an associated customer or not.

The formulation we propose here, F_{xyz}, is based on the F_{xy+} and F_{xz} formulations presented in Ávila et al. (2017). This new formulation has more variables than F_{xy+} and F_{xz} but, as it will be seen in Section 5, they are useful in the exact solution of the DC-CEARP. The formulation F_{xyz} uses the following variables:

$x^k_{ij} =$ number of times that the vehicle k traverses arc $(i,j) \in A$,

$$y^{kc}_{ij} = \begin{cases} 1, & \text{if the customer } c \text{ is serviced by vehicle } k \text{ while traversing arc } (i,j) \in A_R \\ 0, & \text{otherwise.} \end{cases}$$

$$z^k_c = \begin{cases} 1, & \text{if the customer } c \text{ is serviced by vehicle } k \\ 0, & \text{otherwise.} \end{cases}$$

The DC-CEARP can be formulated as

$$\text{Minimize} \sum_{k\in\mathbb{K}} \sum_{(i,j)\in A} \sum d_{ij} x^k_{ij}$$

s.t. :

$$\sum_{(i,j)\in A} d_{ij} x^k_{ij} \le D_{max} \quad \forall k \in \mathbb{K} \tag{1}$$

$$x^k(\delta^+(i)) = x^k(\delta^-(i)) \quad \forall i \in V, \ \forall k \in \mathbb{K} \tag{2}$$

$$\sum_{k\in\mathbb{K}} \sum_{(i,j)\in H_c} y^{kc}_{ij} = 1 \quad \forall c \in \mathbb{H} \tag{3}$$

$$x^k_{ij} \ge y^{kc}_{ij} \quad \forall (i,j) \in A_R, \ \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K} \tag{4}$$

$$\sum_{(i,j)\in H_c} y^{kc}_{ij} = z^k_c \quad \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K} \tag{5}$$

$$x^k(\delta^+(S)) \ge z^k_c - x^k(H_c \cap A(V \setminus S)) \quad \forall S \subset V\setminus\{1\}, \ \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K} \tag{6}$$

$$x^k_{ij} \ge 0 \text{ and integer } \forall (i,j) \in A, \ \forall k \in \mathbb{K} \tag{7}$$

$$z^k_c \in \{0,1\} \quad \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K} \tag{8}$$

$$y^{kc}_{ij} \in \{0,1\} \quad \forall (i,j) \in A_R, \ \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K} \tag{9}$$

Inequalities (1) limit the maximum length of each vehicle route. Constraints (2) are the well known symmetry equations. Inequalities (3) force each customer to be serviced exactly from one arc and with one vehicle, and inequalities (4) say that if a vehicle services a required arc then it has to traverse it. The relation between the y^{kc}_{ij} and z^k_c variables is given by Eq. (5). The connectivity of each route is guaranteed by inequalities (6). They are valid because, if vehicle k does not service customer c, $z^k_c = 0$ and the inequality is trivially satisfied. Otherwise, if vehicle k services customer c by traversing an arc in $H_c \cap A(V\setminus S)$, then it does not need to traverse the cutset $\delta(S)$ and the inequality is also trivially satisfied. Only if vehicle k services customer c by traversing an arc not in $H_c \cap A(V\setminus S)$ (hence, traversing an arc in $\delta(S)$ or in $A(S)$), the vehicle has to traverse $\delta(S)$ and, therefore, the inequality is satisfied. Note that there is an exponential number of such inequalities. Finally, (7)–(9) are the non-negativity and integrality constraints.

Note that the coefficients in the objective function and those in inequalities (1) do not necessarily have to be the same. We have set the same coefficients for the sake of simplicity and because we think of them as distances associated with a time that the routes should not exceed because they may correspond, for example, to drivers' working hours.

3. Valid inequalities

In this section we introduce some inequalities that are valid for the DC-CEARP and that will strengthen the linear relaxation of the formulation.

3.1. More connectivity inequalities

Besides the connectivity inequalities (6) in the formulation, involving variables x and z, other connectivity inequalities are presented in what follows.

In Ávila et al. (2017), the following connectivity inequalities were introduced:

$$x^k(\delta^+(S)) \ge 1 - x^k(H_c \cap A(V \setminus S)) - \sum_{k'\ne k} x^{k'}(H_c), \quad \forall S \subset V \setminus \{1\}, \quad \forall c \in \mathbb{H}, \forall k \in \mathbb{K}. \tag{10}$$

These inequalities ensure that, if no vehicle other than k traverses the arcs in H_c (thus it cannot service customer c), and vehicle k does not traverse any arcs in $H_c \cap A(V\setminus S)$, then vehicle k has to traverse the cutset $(V\setminus S, S)$ in order to service this customer. They are called *disaggregate* connectivity inequalities because they refer to a single vehicle. For each subset $\Omega \subset \mathbb{K}$ of $|\Omega| \ge 2$ vehicles, the following Ω-aggregate connectivity inequalities are valid

$$\sum_{k\in\Omega} x^k(\delta^+(S)) \ge 1 - \sum_{k\in\Omega} x^k(H_c \cap A(V\setminus S)) - \sum_{k'\notin\Omega} x^{k'}(H_c), \quad \forall S \subset V \setminus \{1\}, \ \forall c \in \mathbb{H}. \tag{11}$$

In the case $\Omega = \mathbb{K}$ the aggregate connectivity inequalities are:

$$\sum_{k\in\mathbb{K}} x^k(\delta^+(S)) \ge 1 - \sum_{k\in\mathbb{K}} x^k(H_c \cap A(V\setminus S)), \tag{12}$$

for any subset $S \subseteq V\setminus\{1\}$ and any customer $c \in \mathbb{H}$.

If we consider also y^{kc}_{ij} variables, we have a different family of connectivity inequalities (see Ávila et al., 2017):

$$x^k(\delta^+(S)) \ge \sum_{(i,j)\in H_c \setminus A(V\setminus S)} y^{kc}_{ij}, \quad \forall S \subset V\setminus\{1\}, \ \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K} \tag{13}$$

Note that, if vehicle k services customer c using an arc in $H_c \setminus A(V\setminus S)$, then the vehicle has to traverse $\delta(S)$.

Unlike for inequalities (10), the Ω-aggregate and aggregate versions of connectivity inequalities (6) and (13) are just the sum of the corresponding disaggregate inequalities and, therefore, they are dominated.

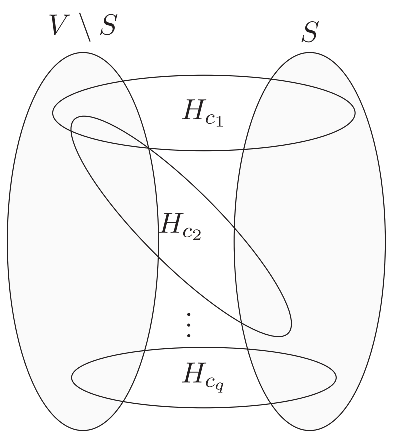
3.2. Parity inequalities

Parity inequalities are based on the fact that a vehicle crosses any cutset an even (or zero) number of times. The parity inequalities for the DC-CEARP described in what follows are different from those in other arc routing problems because they are related not only to the arcs in the cutset but also to the sets H_c. Four different families of *parity inequalities* were presented in Ávila et al. (2017). From them, the two stronger ones are presented in what follows.

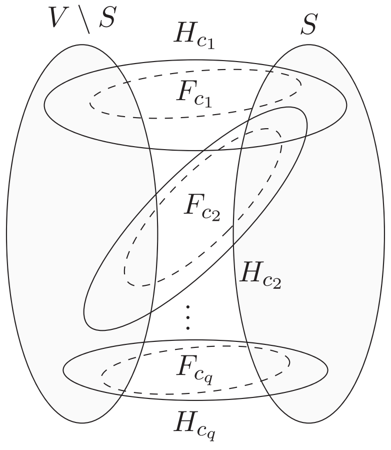
The first family uses only x variables. Given a vehicle k, let $S \subset V$ and consider a subset of customers $\{c_1, c_2, \dots, c_q\}$, with $q \ge 3$ and odd, such that $H_{c_i} \cap H_{c_j} \cap \delta(S) = \emptyset$ and $H_{c_i} \cap \delta(S) \ne \emptyset, \forall c_i, i = 1, \dots, q$ (see Fig. 1(a)). In Ávila et al. (2017), it is proved that the following parity inequalities are valid for the DC-CEARP:

$$x^k(\delta(S)) \ge \sum_{i=1}^{q} \Big(1 - 2\sum_{k'\ne k} x^{k'}(H_{c_i}) - 2x^k(H_{c_i} \setminus \delta(S))\Big) + 1. \tag{14}$$

Note that, if no other vehicle $k' \ne k$ services customer c_i (i.e., $\sum_{k'\ne k} x^{k'}(H_{c_i}) = 0$) and vehicle k does not traverse any edge of customer c_i that is not in the cutset (i.e., $x^k(H_{c_i} \setminus \delta(S)) = 0$), then k



(a) *Parity Inequalities* (14).



(b) *Parity Inequalities* (15).

Fig. 1. Structure of the parity inequalities for the DC-CEARP.

traverses at least an arc in $H_{c_i} \cap \delta(S)$. Extending the previous argument to the q customers in $\delta(S)$, vehicle k has to traverse at least q times $\delta(S)$ and, since q is an odd number, it has to go through the cutset one more time.

The second set of inequalities use the x_{ij}^k and the y_{ij}^{kc} variables. Consider now the set of arc subsets $\mathcal{F} = \{F_{c_1}, F_{c_2}, \ldots, F_{c_q}\}$, with $q \geq 3$ and odd, satisfying $F_{c_i} \subseteq H_{c_i} \cap \delta(S)$ and $F_{c_i} \cap F_{c_j} = \emptyset, \forall c_i, c_j$ (see Fig. 1(b)). Then, the following parity inequalities are valid for the DC-CEARP:

$$x^k(\delta(S)) \geq \sum_{i=1}^{q} \left(2y^{kc_i}(F_{c_i}) - 1\right) + 1. \qquad (15)$$

In this case, note that if vehicle k services each customer c_i from an arc in F_{c_i}, then it has to traverse $\delta(S)$ at least q times. Again, since q is an odd number, the number of traversals should be at least $q+1$.

Besides the right-hand side of the inequalities, there is a difference between the conditions satisfied by the customers in the parity inequalities above. As it is depicted in Fig. 1, two customers c_1 and c_2 satisfying $H_{c_1} \cap H_{c_2} \cap \delta(S) \neq \emptyset$ cannot be considered for inequality (14), but they can for inequality (15) if F_{c_1} and F_{c_2} are chosen such that $F_{c_1} \cap F_{c_2} = \emptyset$. Nevertheless, we want to point out that the greater the sets F_{c_i}, the stronger inequalities (15). In particular, if $F_{c_i} = H_{c_i} \cap \delta(S)$, for all $i = 1, \ldots, q$, and $F_{c_i} \cap F_{c_j} = \emptyset$ for all $i \neq j;\, i, j = 1, \ldots, q$, we obtain the strongest inequality.

By comparing both kind of inequalities, it can be seen that none of them dominates the other in all the cases. Hence, in the branch-and-cut algorithm we will use both families of parity inequalities (14) and (15).

Finally, given a set of vehicles $\Omega = \{k_1, \ldots, k_P\}$, $2 \leq P \leq K$, we have the following *Ω-aggregate parity inequalities*:

$$\sum_{k\in\Omega} x^k(\delta^+(S)) \geq q + 1 - 2\sum_{i=1}^{q}\left(\sum_{k'\notin\Omega} x^{k'}(H_{c_i}) + \sum_{k\in\Omega} x^k(H_{c_i} \setminus \delta(S))\right). \qquad (16)$$

$$\sum_{k\in\Omega} x^k(\delta^+(S)) \geq \sum_{i=1}^{q}\left(\sum_{k\in\Omega} 2y^{kc_i}(F_{c_i}) - 1\right) + 1. \qquad (17)$$

It can be seen that if $\Omega = \mathbb{K}$ and $F_{c_i} = H_{c_i}, \forall c_i$ (hence $H_{c_i} \setminus \delta(S) = \emptyset$ holds), inequalities (16) and (17) reduce to the following aggregate parity inequality:

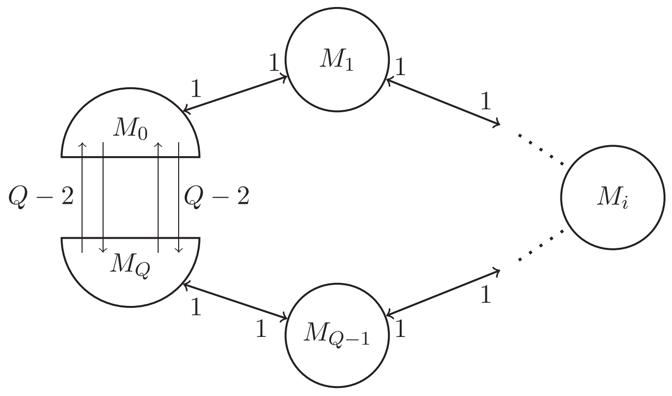


Fig. 2. Standard K-C basic structure.

$$\sum_{k\in\mathbb{K}} x^k(\delta^+(S)) \geq q + 1.$$

3.3. K-C inequalities

K-C inequalities were introduced in Corberán and Sanchis (1994) for the undirected Rural Postman Problem. Beyond the connectivity and parity inequalities described before, the K-C inequalities try to make connectivity and parity conditions satisfied simultaneously on a partition of the vertex set that is more complex than the two shores of the cutsets $(S, V\backslash S)$ used in connectivity and parity inequalities.

The name of this family of inequalities is motivated by the number of sets into which V is partitioned, which is usually denoted by K. To avoid confusion with the number of vehicles, in what follows we use the letter Q instead.

All the versions of the K-C inequalities are based on a structure (see Fig. 2) defined by a partition of the set of vertices V into $Q+1$ subsets, $M_0, M_1, \ldots, M_{Q-1}, M_Q$, and a set of coefficients for the arcs or edges of the graph. For each $(i, j) \in A$, we define

$$\alpha_{ij} = \begin{cases} Q-2, & \text{if } (i,j) \in (M_0, M_Q) \\ |r-s|, & \text{if } (i,j) \in (M_r, M_s),\ \{r,s\} \neq \{0, Q\} \\ 0, & \text{otherwise.} \end{cases} \qquad (18)$$

Let us call *external arcs* those joining two consecutive sets M_r and M_{r+1}, and *internal arcs* to those joining two sets M_r and M_s

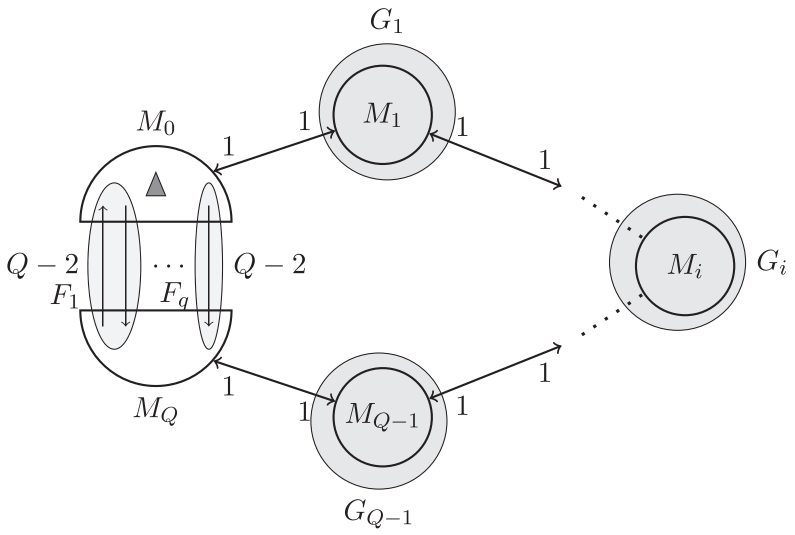


Fig. 3. Standard disaggregate K-C inequality for the DC-CEARP. Depot is represented by a triangle.

with $|r-s|>1$ and $\{r, s\}\neq\{0, Q\}$. Note that all the external arcs have coefficient 1, while the coefficient of an internal arc from M_r to M_s (not shown in Fig. 2) is equal to the cost of the shortest path using the coefficients of the external arcs. Finally, the coefficient of the arcs in (M_0, M_Q) is $Q-2$. It is known (see Corberán & Sanchis, 1994 and Ávila et al., 2017) that any vector $x\in\mathbb{Z}^{|A|}$ representing a tour traversing at least an even number $q\geq 2$ of times the arcs in (M_0, M_Q), and visiting at least once each node set $M_0\cup M_Q$, $M_1, \ldots, M_{Q-1}$, satisfies the following K-C inequality:

$$\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}\geq (Q-2)q+2(Q-1). \tag{19}$$

Inequality (19) is valid for any ARP in which all the tours x must traverse q times the arcs in (M_0, M_Q) and visit all the node sets $M_0\cup M_Q$, $M_1, \ldots, M_{Q-1}$. As an example, this is the case of the ARPs with one single vehicle when there are q required arcs in (M_0, M_Q) and some required arcs in all the sets $M_1,\ldots,M_{Q-1}$. In an ARP with several vehicles, such as the DC-CEARP studied here, it is usual that single vehicles are not obliged to traverse all the required arcs (or, therefore, to visit all the nodes), but only those arcs that are serviced by it. Thus, the K-C inequalities for the DC-CEARP presented in this section have the same left-hand side (LHS) as inequality (19), but the right-hand side (RHS) must include variables y_{ij}^{kc} that define the service of a customer by a vehicle from an arc, in such a way that, when the vehicle k satisfies the above conditions, the RHS of the inequality takes value $(Q-2)q+2(Q-1)$.

3.3.1. Disaggregate K-C inequalities

Consider a partition of the set of vertices V into Q subsets $\{M_0\cup M_Q, M_1,\ldots,M_{Q-1}\}$, with $Q\geq 3$, and the set of coefficients α_{ij} given in (18) (see Fig. 3).

Let us consider a family of arc subsets $\mathcal{F}=\{F_1, F_2,\ldots,F_q\}$, with $q\geq 2$ and even, satisfying (see Fig. 3):

- $F_i\neq\emptyset\quad\forall i\in\{1,\ldots,q\}$,
- $\exists\ c_i\in\mathbb{H}$ such that $F_i\subseteq H_{c_i}\cap(M_0,M_Q),\quad\forall i\in\{1,\ldots,q\}$,
- $F_i\cap F_j=\emptyset,\quad\forall i,j\in\{1,\ldots,q\},\ i\neq j$.

Furthermore, assume that for each M_j, $j=1,\ldots,Q-1$, either $1\in M_j$ or the set of arcs $G_j=H_{c_j}\cap\big(A(M_j)\cup\delta(M_j)\big)$ is nonempty, for some $c_j\in\mathbb{H}$. Note that we cannot assume $G_{j_1}\cap G_{j_2}=\emptyset$ because $\delta(M_{j_1})$ and $\delta(M_{j_2})$ are not necessarily disjoint sets. We define the *disaggregate K-C inequality* associated with a vehicle k as:

$$\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}^k\geq (Q-2)\sum_{i=1}^{q}\Big(2y^{kc_i}(F_i)-1\Big)+\sum_{j=1}^{Q-1}2y^{kc_j}(G_j), \tag{20}$$

if the depot is in $M_0\cup M_Q$, and

$$\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}^k\geq (Q-2)\sum_{i=1}^{q}\Big(2y^{kc_i}(F_i)-1\Big)+\sum_{\substack{j=1\\ j\neq l.}}^{Q-1}2y^{kc_j}(G_j)+2, \tag{21}$$

if $1\in M_l$ with $l\notin\{0, Q\}$.

Note 1. If $Q=2$, then inequality (20) is exactly the connectivity constraint (6) associated with set $S=M_1$.

Theorem 1. *For each vehicle k, disaggregate K-C inequalities (20) and (21) are valid for the DC-CEARP.*

Proof. See Appendix A.1. □

3.3.2. Ω-aggregate K-C inequalities

Here we present the K-C inequalities associated with any subset of vehicles $\Omega\subseteq\mathbb{K}$. Note that, inequality (20) can be written as

$$\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}^k-(Q-2)\sum_{i=1}^{q}\Big(2y^{kc_i}(F_i)\Big)-\sum_{j=1}^{Q-1}2y^{kc_j}(G_j)\geq-(Q-2)q, \tag{22}$$

where the values for coefficients α_{ij} are given in (18).

If we consider a subset of vehicles $\Omega\subseteq\mathbb{K}$ and we add the $|\Omega|$ corresponding disaggregate K-C inequalities we obtain an inequality that is obviously valid for the DC-CEARP, but it is not interesting for the problem, since it is dominated:

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}^k-(Q-2)\sum_{k\in\Omega}\sum_{i=1}^{q}\Big(2y^{kc_i}(F_i)\Big)-\sum_{k\in\Omega}\sum_{j=1}^{Q-1}2y^{kc_j}(G_j)$$
$$\geq-|\Omega|(Q-2)q$$

However, by changing the RHS from $-|\Omega|(Q-2)q$ to $-(Q-2)q$, we obtain new and stronger inequalities (except when RHS=0, i.e., when $Q=3$, $q=2$ and the depot is not in $M_0\cup M_Q$). Specifically, given a partition $\{M_0\cup M_Q, M_1,\ldots,M_{Q-1}\}$, $Q\geq 3$, with the

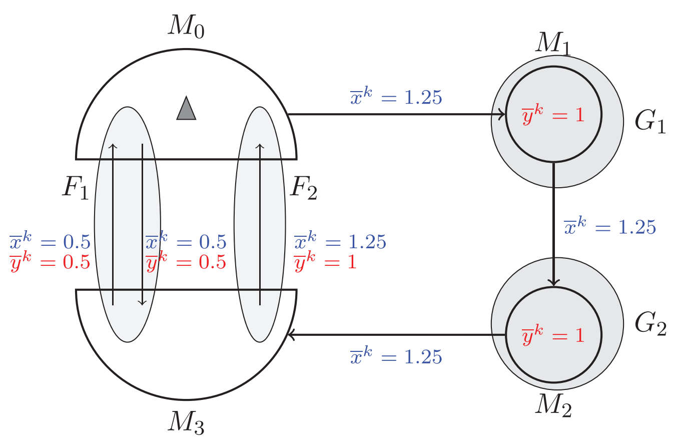


Fig. 4. A fractional solution for vehicle k not cut off by a disaggregate K-C inequality.

corresponding set of coefficients α, a set $\mathcal{F} = \{F_1, F_2, \dots, F_q\}$ ($q \geq 2$ and even) and some sets G_j as above, and given a subset of vehicles Ω, we define the *Ω-aggregate K-C inequality* as

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\alpha_{ij}\, x_{ij}^k \geq (Q-2)\sum_{i=1}^{q}\left(\sum_{k\in\Omega}2y^{kc_i}(F_i)-1\right)+\sum_{j=1}^{Q-1}\sum_{k\in\Omega}2y^{kc_j}(G_j) \tag{23}$$

if the depot $1 \in M_0 \cup M_Q$, and

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\alpha_{ij}\, x_{ij}^k \geq (Q-2)\sum_{i=1}^{q}\left(\sum_{k\in\Omega}2y^{kc_i}(F_i)-1\right) + \sum_{\substack{j=1\\ j\neq l.}}^{Q-1}\sum_{k\in\Omega}2y^{kc_j}(G_j)+2 \tag{24}$$

if $1 \in M_l$, with $l \notin \{0, Q\}$.

Theorem 2. *Given a set of vehicles $\Omega \subseteq \mathbb{K}$, the Ω-aggregate K-C inequalities (23) and (24) are valid for the DC-CEARP.*

Proof. See Appendix A.2. □

3.4. K-C$_{02}$ inequalities

K-C$_{02}$ inequalities are a variant of the K-C inequalities that take into account the asymmetry of the costs associated with the direction of traversal. In some ARPs the K-C$_{02}$ inequalities are dominated by the standard K-C inequalities. This is not the case for the DC-CEARP. For example, consider the fractional DC-CEARP solution $(\overline{x}^k, \overline{y}^k)$ depicted in Fig. 4.

It can be seen that this solution satisfies all the connectivity inequalities (6) and it also satisfies the K-C inequality (20) corresponding to this structure:

$$\sum_{(i,j)\in A}\alpha_{ij}\,\overline{x}_{ij}^k = (0.5+0.5+1.25)+(1.25+1.25+1.25) = 6, \quad \text{and}$$

$$(Q-2)\sum_{i=1}^{q}\left(2\overline{y}^{kc_i}(F_i)-1\right)+\sum_{i=1}^{q}2\overline{y}^{kc_j}(G_j) = \left(2(0.5+0.5)-1\right) + \left(2\times 1-1\right)+2\times 1+2\times 1 = 6.$$

We will see that the disaggregate K-C$_{02}$ inequalities that we describe in what follows do cut off this solution.

3.4.1. Disaggregate K-C$_{02}$ inequalities

Consider a partition of the set of vertices V into Q subsets $\{M_0 \cup M_Q, M_1, \dots, M_{Q-1}\}$, with $Q \geq 2$, and the following set of coefficients (see Fig. 5). For each $(i, j) \in A$,

$$\beta_{ij} = \begin{cases} Q-1, & \text{if } (i,j) \in (M_0, M_Q) \\ s-1, & \text{if } (i,j) \in (M_0 : M_s),\ 1 \leq s \leq Q-1 \\ s+1, & \text{if } (i,j) \in (M_s : M_0),\ 1 \leq s \leq Q-1 \\ |r-s|, & \text{if } (i,j) \in (M_r, M_s),\ 1 \leq r,s \leq Q \\ 0, & \text{otherwise.} \end{cases}$$

Let us also consider a family of arc subsets $\mathcal{F} = \{F_1, F_2, \dots, F_q\}$, and the arc sets G_j satisfying the same conditions as for the K-C inequalities. Note that now we have $Q \geq 2$ (see Note 2 below). We define the *disaggregate K-C$_{02}$ inequalities* associated with a vehicle k as:

$$\sum_{(i,j)\in A}\beta_{ij}\, x_{ij}^k \geq (Q-1)\sum_{i=1}^{q}\left(2y^{kc_i}(F_i)-1\right)+\sum_{j=1}^{Q-1}2y^{kc_j}(G_j), \tag{25}$$

if the depot is in $M_0 \cup M_Q$, and

$$\sum_{(i,j)\in A}\beta_{ij}\, x_{ij}^k \geq (Q-1)\sum_{i=1}^{q}\left(2y^{kc_i}(F_i)-1\right)+\sum_{\substack{j=1\\ j\neq l.}}^{Q-1}2y^{kc_j}(G_j)+2, \tag{26}$$

if $1 \in M_l$ with $l \notin \{0, Q\}$.

Theorem 3. *For each vehicle k, K-C$_{02}$ inequalities (25) and (26) are valid for the DC-CEARP.*

Proof. The proof is similar to that of Theorem 1 and is omitted here for the sake of brevity. □

Let us now check that the K-C$_{02}$ inequality cuts off the fractional solution $(\overline{x}^k, \overline{y}^k)$ depicted in Fig. 4:

$$\sum_{(i,j)\in A}\beta_{ij}\,\overline{x}_{ij}^k = 2(0.5+0.5+1.25)+0\times 1.25 + 1\times 1.25 + 1\times 1.25 = 7,$$

while

$$(Q-1)\sum_{i=1}^{q}\left(2\overline{y}^{kc_i}(F_i)-1\right)+\sum_{i=1}^{q}2\overline{y}^{kc_j}(G_j) = 2\Big(2(0.5+0.5)-1+2\times 1-1)\Big)+2\times 1+2\times 1 = 8.$$

Note 2. Unlike the standard K-C inequalities, the K-C$_{02}$ inequalities with $Q = 2$ are not equivalent to any other known inequality.

3.4.2. Ω-aggregate K-C$_{02}$ inequalities

Given a partition $\{M_0 \cup M_Q, M_1, \dots, M_{Q-1}\}$, $Q \geq 2$, with the corresponding set of coefficients β, a set $\mathcal{F} = \{F_1, F_2, \dots, F_q\}$ ($q \geq 2$ and even), some sets G_j as above, and given a subset of vehicles $\Omega \subseteq \mathbb{K}$, we define the *$\Omega$-aggregate K-C$_{02}$ inequality* as

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\beta_{ij}\, x_{ij}^k \geq (Q-1)\sum_{i=1}^{q}\left(\sum_{k\in\Omega}2y^{kc_i}(F_i)-1\right)+\sum_{j=1}^{Q-1}\sum_{k\in\Omega}2y^{kc_j}(G_j), \tag{27}$$

if the depot $1 \in M_0 \cup M_Q$, and

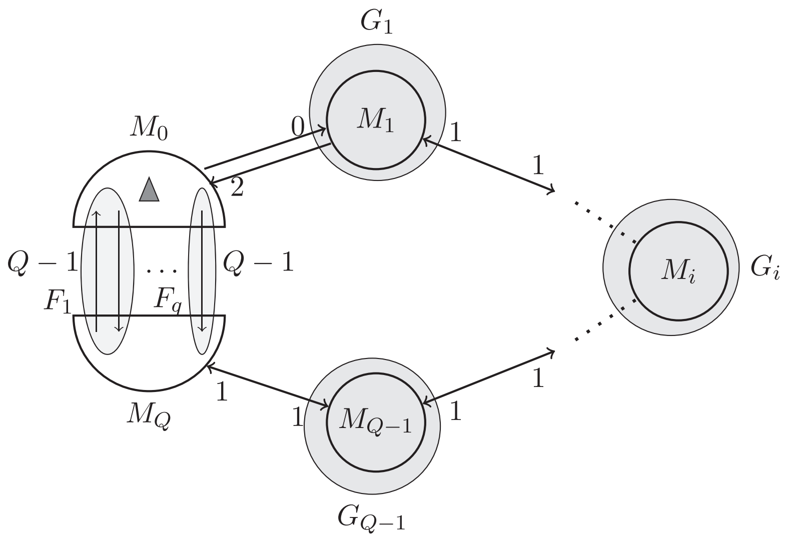


Fig. 5. Disaggregate K-C$_{02}$ inequalities for the DC-CEARP.

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\beta_{ij}\,x_{ij}^k \ge (Q-1)\sum_{i=1}^{q}\Big(\sum_{k\in\Omega}2y^{kc_i}(F_i)-1\Big) + \sum_{\substack{j=1\\ j\neq l.}}^{Q-1}\sum_{k\in\Omega}2y^{kc_j}(G_j)+2, \qquad (28)$$

if $1\in M_l$, with $l\notin\{0,Q\}$.

Theorem 4. *Given a set of vehicles $\Omega\subseteq\mathbb{K}$, the Ω-aggregate K-C inequalities (27) and (28) are valid for the DC-CEARP.*

Proof. The proof is similar to that of Theorem 2 and is omitted here for the sake of brevity. □

3.5. Path-Bridge inequalities

Path-Bridge inequalities are a generalization of the K-C inequalities introduced in Letchford (1997) for the undirected General Routing Problem and are inspired by the path inequalities introduced in Cornuèjols, Fonlupt, and Naddef (1985) for the Graphical Traveling Salesman Problem.

As K-C, Path-Bridge inequalities try that connectivity and parity conditions are satisfied simultaneously on a given partition of the vertex set V. They are based on a structure (see Fig. 6) with two sets M_0, M_Z, a number $P\ge 1$ of ‘paths’ between M_0 and M_Z, and a number $B\ge 0$ of required arcs in (M_0, M_Z) forming the ‘bridge’, with $P+B\ge 3$ being an odd number. It can be noted that K-C inequalities described in Section 3.3 are a particular case of the Path-Bridge inequalities when $P=1$ and $B\ge 2$ and even.

3.5.1. Disaggregate path-bridge inequalities

Given two integers $P\ge 1$, $B\ge 0$ such that $P+B\ge 3$ is an odd number, consider the partition of V into the subsets $\{M_0, M_Z, \{M_r^t\}_{r=1,\dots,n_t}^{t=1,\dots,P}\}$, where $n_1, n_2, \dots, n_t$ are integer numbers, $n_i\ge 2$, and consider the following coefficients (see Fig. 6). For each $(i,j)\in A$,

$$\alpha_{i,j}=\begin{cases}1, & \text{if } (i,j)\in(M_0,M_Z)\\ \frac{|r-s|}{n_t-1}, & \text{if } (i,j)\in(M_r^t,M_s^t),\, t\in\{1,\dots,P\},\ r,s\in\{0,1,\dots,n_t+1\}\\ \frac{1}{n_t-1}+\frac{1}{n_u-1}+\left|\frac{r-1}{n_t-1}-\frac{s-1}{n_u-1}\right|, & \text{if } (i,j)\in(M_r^t,M_s^u),\, t\neq u,\ r\in\{1,\dots,n_t\},\ s\in\{1,\dots,n_u\}\\ 0, & \text{otherwise.}\end{cases}$$

Let us consider a family of arc subsets $\mathcal{F}=\{F_1,F_2,\dots,F_B\}$ satisfying:

- $F_i\neq\emptyset \quad \forall i\in\{1,\dots,B\}$,
- $\exists\ c_i\in\mathbb{H}$ such that $F_i\subseteq H_{c_i}\cap(M_0,M_Z) \quad \forall i\in\{1,\dots,B\}$,
- $F_i\cap F_j=\emptyset, \quad \forall i\neq j\in\{1,\dots,B\}$.

Furthermore, assume that for each M_r^t, $t=1,\dots,P$, $r=1,\dots,n_t$, either $1\in M_r^t$ or it exists a set of arcs G_r^t such that $\emptyset\neq G_r^t\subseteq H_{c_j}\cap A_R(M_r^t\cup\delta(M_r^t))$ for a $c_j\in\mathbb{H}$. We define the *disaggregate Path-Bridge inequality* associated with vehicle k as:

$$\sum_{(i,j)\in A}\alpha_{ij}\,x_{ij}^k \ge \sum_{i=1}^{B}\Big(2y^{kc_i}(F_i)-1\Big)+\sum_{t=1}^{P}\sum_{r=1}^{n_t}\frac{2y^{kc_j}(G_r^t)}{n_t-1}-P+1, \qquad (29)$$

if the depot $1\in M_0\cup M_Z$, and

$$\sum_{(i,j)\in A}\alpha_{ij}\,x_{ij}^k \ge \sum_{i=1}^{B}\Big(2y^{kc_i}(F_i)-1\Big)+\sum_{\substack{t=1\\ t\neq t_0}}^{P}\sum_{r=1}^{n_t}\frac{2y^{kc_j}(G_r^t)}{n_t-1} + \sum_{\substack{r=1\\ r\neq r_0}}^{n_{t_0}}\frac{2y^{kc_j}(G_r^{t_0})}{n_{t_0}-1}+\frac{2}{(n_{t_0}-1)}-P+1, \qquad (30)$$

if the depot $1\in M_{r_0}^{t_0}$ (different from M_0 and M_Z).

Theorem 5. *For each vehicle k, disaggregate Path-Bridge inequalities (29) and (30) are valid for the DC-CEARP.*

Proof. See Appendix A.3. □

If we multiply the Path-Bridge inequalities (29) and (30) by $\prod_{t=1}^{P}(n_t-1)$, all the coefficients become integer. When $P=2$ (and,

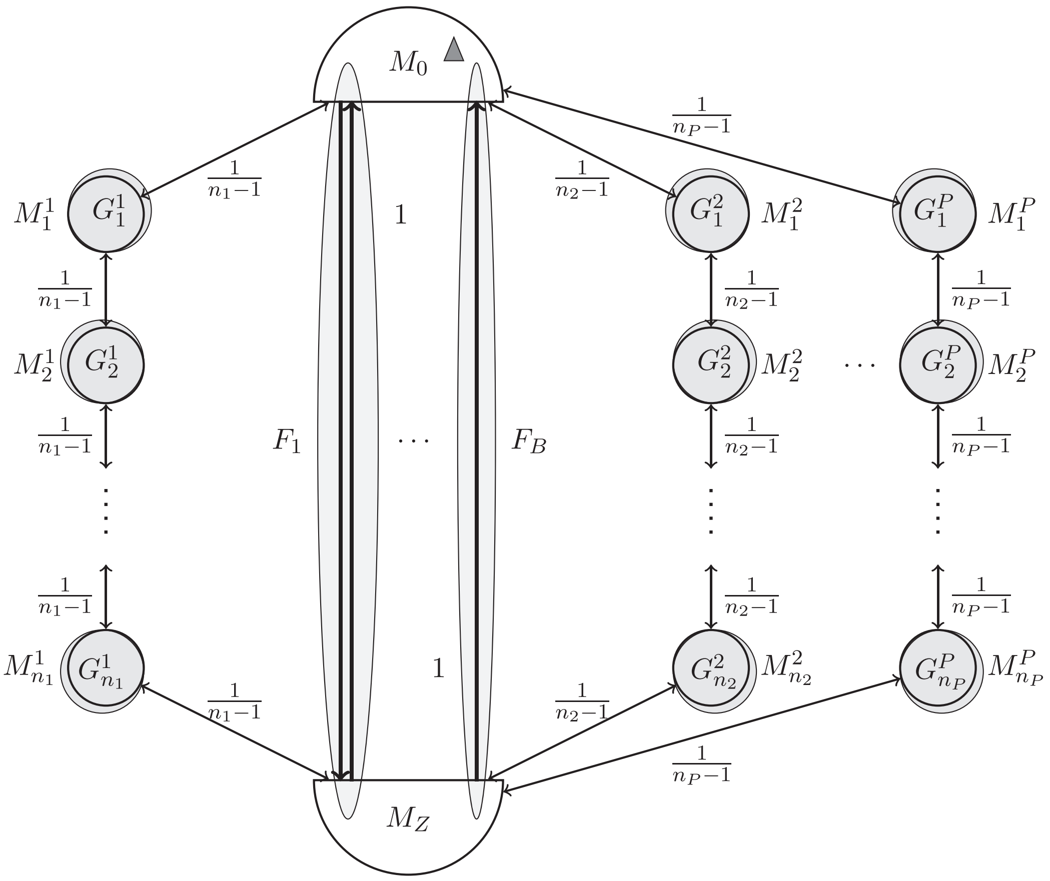


Fig. 6. Standard Path-Bridge for the DC-CEARP.

hence, B is an odd number), the inequality is called *2-Path-Bridge inequality* and can be written as:

$$\sum_{(i,j)\in A}(n_1-1)(n_2-1)\alpha_{ij}\ x_{ij}^k \geq \sum_{i=1}^{B} 2(n_1-1)(n_2-1)y^{kc_i}(F_i)$$
$$+\sum_{j=1}^{n_1} 2(n_2-1)y^{kc_j}(G_j^1)$$
$$+\sum_{j=1}^{n_2} 2(n_1-1)y^{kc_j}(G_j^2)-(B+1)(n_1-1)(n_2-1), \tag{31}$$

when the depot $1 \in M_0 \cup M_Z$. If the depot is, for example, in a node j_0 of the path $t = 1$ $(1 \in M_{j_0}^1)$, the resulting inequality is

$$\sum_{(i,j)\in A}(n_1-1)(n_2-1)\alpha_{ij}\ x_{ij}^k \geq \sum_{i=1}^{B} 2(n_1-1)(n_2-1)y^{kc_i}(F_i)+2(n_2-1)$$
$$+\sum_{\substack{j=1\\ j\neq j_0}}^{n_1} 2(n_2-1)y^{kc_j}(G_j^1)+\sum_{j=1}^{n_2}2(n_1-1)y^{kc_j}(G_j^2)-(B+1)(n_1-1)(n_2-1). \tag{32}$$

3.5.2. Ω-aggregate Path-Bridge inequalities

Given P, B, a partition $\{M_0, M_Z, \{M_r^t\}_{r=1,\dots,n_t}^{t=1,\dots,P}\}$, its corresponding set of coefficients α_{ij}, and the families of arcs F_i and G_r^t, as in the previous section, we define the *Ω-aggregate Path-Bridge inequality* associated with a subset $\Omega \subseteq \mathbb{K}$ of vehicles as:

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}^k \geq \sum_{i=1}^{B}\Big(\sum_{k\in\Omega}2y^{kc_i}(F_i)-1\Big)$$
$$+\sum_{k\in\Omega}\sum_{s=1}^{P}\Big(\sum_{q=1}^{n_t}\frac{2y^{kc_j}(G_r^t)}{n_t-1}\Big)-P+1, \tag{33}$$

if the depot $1 \in M_0 \cup M_Z$, and

$$\sum_{k\in\Omega}\sum_{(i,j)\in A}\alpha_{ij}\ x_{ij}^k$$
$$\geq \sum_{i=1}^{B}\Big(\sum_{k\in\Omega}2y^{kc_i}(F_i)-1\Big)+\sum_{k\in\Omega}\sum_{\substack{t=1\\ t\neq t_0}}^{P}\sum_{r=1}^{n_t}\frac{2y^{kc_j}(G_r^t)}{n_t-1}$$
$$+\sum_{k\in\Omega}\sum_{\substack{r=1\\ r\neq r_0}}^{n_{t_0}}\frac{2y^{kc_j}(G_r^{t_0})}{n_{t_0}-1}+\frac{2}{(n_{t_0}-1)}-P+1, \tag{34}$$

if the depot $1 \in M_{r_0}^{t_0}$ (different from M_0 and M_Z).

Theorem 6. *Given a set of vehicles $\Omega \subseteq \mathbb{K}$, the Ω-aggregate Path-Bridge inequalities (33) and (34) are valid for the DC-CEARP.*

Proof. The proof is similar to that of Theorem 2 and is omitted here for the sake of brevity. □

Note 3 (Path-Bridge$_{02}$ inequalities). An asymmetric version of Path-Bridge inequalities, called Path-Bridge$_{02}$ inequalities, is proposed in Corberán, Romero, and Sanchis (2003) for the Mixed General Routing Problem. *Inequalities based on the same idea can also be proposed for the DC-CEARP. However, not all their coefficients can be easily determined, since the coefficients of the variables associated with arcs between nodes of different paths must be computed by sequential lifting for each particular Path-Bridge structure. This process* *is involved and, in addition, the obtained coefficients depend on the ordering in which arcs are considered. For this reason, its separation has never been implemented and, therefore, these inequalities are not studied here.*

3.6. Max-distance constraints

In the DC-CEARP, the length of each route cannot exceed the maximum distance D_{max}. Based on this constraint, in this section we present several sets of inequalities that we call *max-distance inequalities*.

Let $F^{\mathbb{H}} \subseteq \mathbb{H}$ be a subset of customers. Consider the Close Enough Arc Routing Problem (which considers only one vehicle), defined on graph G and with set of customers $F^{\mathbb{H}}$. Let $\text{CEARP}(F^{\mathbb{H}})$ be its optimal value (or a lower bound of it). If $\text{CEARP}(F^{\mathbb{H}}) > D_{max}$, then the inequalities

$$z_c^k(F^{\mathbb{H}}) \leq |F^{\mathbb{H}}| - 1, \quad \forall k \in \mathbb{K} \tag{35}$$

are valid for the DC-CEARP, because a single vehicle cannot service all the customers in $F^{\mathbb{H}}$.

On the other hand, if S is the set of vertices incident with the arcs in $\cup_{c \in F^{\mathbb{H}}} H_c$ and $1 \notin S$, then at least two different vehicles have to enter S, and the following inequality is also valid for the DC-CEARP

$$\sum_{k \in \mathbb{K}} x^k(\delta^-(S)) \geq 2. \tag{36}$$

However, a DC-CEARP solution in which a vehicle enters S twice, but no other vehicle does, satisfies (36). In order to force two different vehicles to enter S, the following valid inequalities can be used

$$\sum_{k \neq k'} x^k(\delta^-(S)) \geq 1, \quad \forall k' \in \mathbb{K}. \tag{37}$$

The above inequalities, which were proposed in Ávila et al. (2017), can be generalized as follows. For a given set of customers $F^{\mathbb{H}}$, let $\nu(F^{\mathbb{H}})$ be a lower bound on the minimum number of vehicles needed to service $F^{\mathbb{H}}$. Then, a number of vehicles less than $\nu(F^{\mathbb{H}})$ cannot service all the customers in $F^{\mathbb{H}}$. Hence, if $\nu(F^{\mathbb{H}}) \geq 2$, the following inequalities are satisfied by each feasible solution of the DC-CEARP:

$$\sum_{k \in \Omega} z_c^k(F^{\mathbb{H}}) \leq |F^{\mathbb{H}}| - (\nu(F^{\mathbb{H}}) - |\Omega|), \quad \forall\ \Omega \subseteq \mathbb{K},$$
$$1 \leq |\Omega| \leq \nu(F^{\mathbb{H}}) - 1, \tag{38}$$

$$\sum_{k \in \mathbb{K} \setminus \Omega} x^k(\delta^-(S)) \geq \nu(F^{\mathbb{H}}) - |\Omega|, \quad \forall\ \Omega \subseteq \mathbb{K}, \quad 0 \leq |\Omega| \leq \nu(F^{\mathbb{H}}) - 1. \tag{39}$$

Note that, for $\nu(F^{\mathbb{H}}) = 2$, inequalities (38) and (39) are exactly (35) and (36)+(37) above, respectively. For $\nu(F^{\mathbb{H}}) = 3$, we have two sets of inequalities (38):

$$z_c^k(F^{\mathbb{H}}) \leq |F^{\mathbb{H}}| - 2, \quad \forall k \in \mathbb{K}, \quad \text{and} \tag{40}$$

$$z_c^k(F^{\mathbb{H}}) + z_c^{k'}(F^{\mathbb{H}}) \leq |F^{\mathbb{H}}| - 1, \quad \forall k, k' \in \mathbb{K}. \tag{41}$$

3.7. Symmetry breaking inequalities

Let $\{c_1, \ldots, c_L\}$ be any ordering of the set of customers (for example, according to the distances between them and the depot). The following symmetry breaking inequalities (see Fischetti, Salazar-González, & Toth, 1995) are introduced to avoid equivalent solutions:

$$z_{c_1}^1 = 1 \tag{42}$$

$$z_{c_i}^k \leq \sum_{j=1}^{i-1} z_{c_j}^{k-1} \quad k = 3, \ldots, K, \quad i \geq 2 \tag{43}$$

$$z_{c_i}^k = 0 \quad k = i+1, \ldots, K, \quad i = 1, \ldots, L-1 \tag{44}$$

Inequality (42) forces vehicle 1 to service customer c_1. Then, vehicle 2 services the first customer in the ordering not serviced by vehicle 1, and so on. Inequalities (43) state that if a customer c_i is serviced by vehicle k, then at least one 'previous' customer c_j, $j = 1, \ldots, i-1$, has to be serviced by the vehicle $k-1$. Eq. (44) prevents customers $c_i, i = 1, \ldots, L-1$ from being serviced by vehicles with indices larger than i.

4. The branch-and-cut algorithm

In this section, we present a branch-and-cut algorithm for the DC-CEARP. This new algorithm uses some separation procedures from the methods described in Ávila et al. (2017) and incorporates new ones for some of the inequalities described in Ávila et al. (2017) and for the new inequalities presented in this article. Moreover, an upper bound obtained by the matheuristic algorithm proposed in Corberán et al. (2019) is used.

4.1. Separation algorithms

In what follows we describe the separation algorithms that have been used to identify the following types of inequalities that are violated by the current LP solution at any iteration of the cutting plane algorithm: connectivity and parity inequalities, disaggregate and Ω-aggregate K-C and K-C$_{02}$, Path-Bridge inequalities, and max-distance constraints. Section 4.1.5 provides a synopsis of the cutting planes and characteristics of their separation procedures.

4.1.1. Connectivity inequalities

Several separation procedures have been used to separate connectivity inequalities. The first algorithm, $A1$, separates aggregate connectivity inequalities (12). It is based on computing the connected components of the graph induced by the arcs a such that $\sum_{k \in \mathbb{K}} x_a^k \geq \varepsilon$, where ε is a given parameter. For each weakly connected component, its corresponding aggregate connectivity inequality is checked for violation. We try $\varepsilon = 0, 0.25, 0.5, 0.75,$ but a given value is tried only when the previous one did not succeed in finding a violated inequality.

The second heuristic, $A2$, is based on the Gomory–Hu algorithm. It also works on the aggregate graph induced by the sum of the variables corresponding to all the vehicles. If there is a violated (aggregate) connectivity inequality in this graph, it means that there will be a violated (disaggregate) connectivity inequality (6) for at least one vehicle.

Two more separation procedures for connectivity inequalities (13) have been implemented. The first one, $A3$, works like the first algorithm described in this section, but using the graph induced by the arcs of each single vehicle.

The last algorithm, $A4$, is based on the computation, for each vehicle k and each customer c, of the maximum flow on a network containing the arcs for which $x_{ij}^k > 0$ plus an artificial sink and some artificial arcs from the end vertices of the arcs in H_c serviced by vehicle k (i.e. those arcs a such that $y_a^{kc} > 0$) to the sink. The capacity of the arcs is defined as x_{ij}^k for the arcs in the original graph, and as infinity for the artificial ones. The maximum flow from the depot to the sink defines a minimum cutset $(S, V \setminus S)$, with $1 \in V \setminus S$, and the associated connectivity inequality (13) is checked for violation.

4.1.2. Parity inequalities

We have developed several heuristic algorithms to identify violated parity inequalities. They work as follows.

Given a fractional solution, let (x^k, y^k, z^k) be its part corresponding to vehicle k. We build the graph induced by the arcs satisfying $x_a^k - \bar{y}_a^k \geq \varepsilon$, if a is required, and $x_a^k \geq \varepsilon$ otherwise, where $\bar{y}_a^k = \max_{c\in\mathbb{H}}\{y_a^{kc} : a \in H_c\}$, i.e. the maximum value of y_a^{kc} among the customers serviced by arc a. Let $S_1, \dots, S_r$ be the sets of vertices of the weakly connected components of the induced graph. For each cutset $\delta(S_i)$, we now try to select the set of customers $F^{\mathbb{H}} = \{c_1, \dots, c_q\}$ and the corresponding sets of arcs $\mathcal{F} = \{F_{c_1}, \dots, F_{c_q}\}$. Two different strategies have been implemented in order to find these sets.

In *strategy 1* (algorithm $A5$), we create a list of pairs of required arcs and customers (a, c) such that $a \in \delta(S_i) \cap H_c$. We order this list according to the value of y_a^{kc} in a decreasing order. Starting from the first pair (a, c) of the list, we iteratively add c to $F^{\mathbb{H}}$ and a to F_c if neither arc a nor customer c have been previously selected. Once the set of customers has been built, we try to enlarge sets F_c by including each unselected arc a of the list in the set F_c with maximum y_a^{kc}.

Now, for each vehicle k, we calculate $x^k(\delta(S)) - \sum_{i=1}^{q} 2y^{kc_i}(F_{c_i})$. If this value is less than 0, we add k to the set of chosen vehicles Ω.

If $\sum_{k\in\Omega}\sum_{e\in F_c} y_e^{kc} < 0.5$ for some customer $c \in F^{\mathbb{H}}$, this customer is removed from $F^{\mathbb{H}}$. If $|F^{\mathbb{H}}|$ is even, we add or remove one more customer according to the value of $\sum_{k\in\Omega}\sum_{e\in F_c} y_e^{kc}$ in order to make $|F^{\mathbb{H}}|$ odd. For each removed customer c, the arcs that belonged to F_c are studied to see if they can be included in another arc subset of $\mathcal{F}$.

Finally, we check if the corresponding Ω-aggregate parity inequality (17) is violated.

Strategy 2 (algorithm $A6$) considers only the cutsets $\delta(S_i)$ for which $x^k(\delta(S_i) \cap A_R)$ is close to an odd number, i.e. $2n + 0.75 \leq x^k(\delta(S_i) \cap A_R) \leq 2n + 1.25$ for some $n \in \{1, 2, \dots\}$. Let us call $A_{S_i} = \{a \in \delta(S_i) \cap A_R : \sum_{c: a\in H_c} y_a^{kc} > 0\}$ and $\mathbb{H}_{S_i} = \{c \in \mathbb{H} : \sum_{a\in\delta(S_i)\cap A_R} y_a^{kc} > 0\}$, which denote the set of required arcs in cutset $\delta(S_i)$ servicing some customer and the set of customers that are serviced by some arc in the cutset, respectively.

We calculate $\sum_{a\in A_{S_i}} y_a^{kc}$ for all the customers in $\mathbb{H}_{S_i}$ and select the customer c that maximizes this value. This customer is added to $F^{\mathbb{H}}$ and $F_c = H_c \cap A_{S_i}$. Now we update A_{S_i} by $A_{S_i} \setminus F_c$ and $\mathbb{H}_{S_i}$ by $\mathbb{H}_{S_i} \setminus \{c\}$ and repeat the procedure for choosing the following customers until $|F^{\mathbb{H}}| = 2n + 1$.

As in strategy 1, we include in Ω all the vehicles k for which $x^k(\delta(S)) - \sum_{i=1}^{q} 2y^{kc_i}(F_{c_i}) < 0$ and check if the corresponding Ω-aggregate parity inequality (17) is violated.

In order to separate parity inequalities (16) when $|\Omega| = K$, algorithm $A7$ uses a similar procedure with strategy 2 adapted to the graph induced by the arcs satisfying $\sum_{k\in\mathbb{K}} x_a^k - 1 \geq \varepsilon$, if a is required, and $\sum_{k\in\mathbb{K}} x_a^k \geq \varepsilon$ otherwise.

We have also tried an alternative method for selecting the set of customers $F^{\mathbb{H}}$ based on the solution of the following integer program. As before, we define $\mathbb{H}_{S_i}$ as the set of customers that are serviced by some arc in the cutset $\delta(S_i)$. For each customer $c \in \mathbb{H}_{S_i}$, we define a binary variable μ_c that takes value 1 if c is included in $F^{\mathbb{H}}$ and 0 otherwise. Let us define $w_c = \sum_{a\in\delta(S_i)} y_a^{kc}$ for each customer $c \in \mathbb{H}_{S_i}$ and consider two customers c_r, c_s as incompatible if there is an arc $a \in \delta_R(S_i)$ such that $y_a^{kc_r} > 0$ and $y_a^{kc_s} > 0$. Then, we solve the following IP:

$$\text{Maximize} \quad \sum_{c\in\mathbb{H}_{S_i}} w_c\ \mu_c$$

$$s.t.:$$

$$\sum_{c\in\mathbb{H}_{S_i}} \mu_c \equiv odd \tag{45}$$

$$\mu_{c_r} + \mu_{c_s} \leq 1 \quad \forall c_r, c_s \text{ incompatible} \tag{46}$$

$$\mu_c \in \{0, 1\} \quad \forall c \in \mathbb{H}_{S_i} \tag{47}$$

In order to study the performance of the above method, we have compared it with the heuristic procedure for selecting $F^{\mathbb{H}}$. On a sample of 27 randomly selected instances, the IP-based method used, on average, 123.53 seconds per instance to find, on average, 0.22 violated parity inequalities per call, while the heuristic method found, on average, 0.13 parity cuts in 0.46 seconds of computing time. Based on these results, we have decided not to use the IP-based method.

4.1.3. K-C, K-C$_{02}$ and Path-Bridge inequalities

Algorithm $A8$ looks first for the graph structure associated with the disaggregate K-C inequalities (20) and (21). Again, let (x^k, y^k, z^k) be the part corresponding to vehicle k of a given fractional solution. Let G^k be the graph induced by the arcs a with $x_a^k > 0$ and label the depot and the arcs $a \in A_R$ such that $x_a^k \geq \varepsilon$ and $y_a^k \geq x_a^k/2$ as 'required', where ε is a given parameter. We compute the connected components induced by these arcs and the depot. Let us call C_i to these components. We then apply a procedure based on that described in Corberán, Letchford, and Sanchis (2001) for the undirected GRP to obtain the sets $M_0, M_1, M_2, \dots, M_Q$, which consists of, given a component C_i, checking if it is connected to two different components by arcs with $x_a^k > 0$. For such a component, we try to split it in two parts such that each part is connected to a different component. These two parts will be the "seeds" for defining sets M_0 and M_Q. Now we shrink these seeds and the remaining components into a single vertex each and compute a spanning tree by iteratively adding the arc of maximum weight not forming a cycle (and not connecting the seeds). This tree is transformed into a path linking the seeds by (iteratively) shrinking each non-seed vertex with degree one into its (unique) adjacent vertex. If the length of the path is at least 3, the vertices of the path define the seeds for sets $M_0, M_1, \dots, M_Q$. All the vertices of G that do not belong to a set M_i yet are iteratively assigned to a set M_i to which they are adjacent.

Set $\mathcal{F}$ is formed by some sets of 'required' arcs in $M_0 \cup M_Q$ associated with different customers. For each set M_j, $j = 1, \dots, Q-1$, non containing the depot, we define G_j as the set of arcs $G_j = H_{c_j} \cap \left(A(M_j) \cup \delta(M_j)\right)$. If the corresponding K-C inequality for vehicle k is not violated, we try to improve the inequality by shrinking some consecutive sets M_j. Several values for ε have been tried and, after some computational testing, we finally decided to set $\varepsilon = 0.2$.

At this point, a K-C structure has been found, and its corresponding disaggregate K-C inequality for vehicle k can, or not, be violated. Since inequality (20) (if the depot belongs to $M_0 \cup M_Q$, otherwise it would be similar) can be written as (22), we evaluate its left hand side for each vehicle $k' = 1, \dots, K$. Then we include in Ω all the vehicles k' for which this expression is less than 0 and check if the resulting Ω-aggregate K-C inequality (23) or (24) is violated.

Any K-C structure found by algorithm $A8$ is used to look for violated K-C$_{02}$ inequalities (algorithm $A9$). As before, the inequality for each single vehicle k' is checked and those vehicles for which the left hand side is negative are included into Ω. The separation of 2 Path-Bridge inequalities is done with a similar procedure, algorithm $A10$, that is not described here for the sake of brevity.

4.1.4. Max-distance inequalities

Two heuristic algorithms are used to separate max-distance inequalities. The first heuristic, $A11$, is the one described in Ávila

Table 1
Inequalities and separation procedures.

Class	Inequalities	Separ. proc.	Type	Complexity	New?	Reference
Connectivity	(12)	A1	H	$O(\lvert A\rvert\lvert\mathbb{H}\rvert)$	No	Corberán et al. (2001)
	(6)	A2	AE	$O(\lvert V\rvert^3\lvert A\rvert)$	No	Ávila et al. (2017)
	(13)	A3	H	$O(K\lvert A\rvert\lvert\mathbb{H}\rvert)$	No/Yes	Ávila et al. (2017), [TP]
	(13)	A4	AE	$O(K\lvert\mathbb{H}\rvert\lvert V\rvert^2\lvert A\rvert)$	No/Yes	Ávila et al. (2017), [TP]
Parity	(17)	A5	H	$O(K\lvert\mathbb{H}\rvert\lvert V\rvert\lvert A\rvert)$	Yes	[TP]
	(17)	A6	H	$O(K\lvert\mathbb{H}\rvert\lvert V\rvert\lvert A\rvert)$	Yes	[TP]
	(16)	A7	H	$O(\lvert V\rvert\lvert A\rvert\lvert\mathbb{H}\rvert)$	Yes	[TP]
K-C	(20), (21), (23), (24)	A8	H	$O(K\lvert V\rvert^2\lvert A\rvert)$	No/Yes	Corberán et al. (2001), [TP]
K-C$_{02}$	(25)–(28)	A9	H	$O(K\lvert V\rvert^2\lvert A\rvert)$	No/Yes	Corberán et al. (2003), [TP]
2 Path-Bridge	(29), (30), (33), (34)	A10	H	$O(K\lvert V\rvert^2\lvert A\rvert)$	No/Yes	Corberán et al. (2001), [TP]
Max-distance	(36)–(38)	A11	H	$O(\lvert\mathbb{H}\rvert)*$	No	Ávila et al. (2017)
	(36)–(38)	A12	H	$O(K)*$	Yes	[TP]

et al. (2017) for separating inequalities (36). If a violated max-distance constraint (36) is found, at least one of the inequalities (37) is also violated and it is added. Furthermore, the corresponding inequality (38) is also added.

The second heuristic, $A12$, looks for violated inequalities (38). It is designed to cut fractional solutions in which, for a vehicle k, several z_c^k variables take values close to 1 and another one takes a value close to 0.5. It works as follows.

Given a fractional solution associated with vehicle k, (x^k, y^k, z^k), let $\{c_1, c_2, \dots, c_q\}$ be the set of customers such that $z_{c_1}^k \geq z_{c_2}^k \geq \dots \geq z_{c_q}^k \geq 0.5$. We define $F^{\mathbb{H}} = \{c_1, c_2, \dots, c_f\}$, where f is the maximal number such that $z_c^k(F^{\mathbb{H}}) > |F^{\mathbb{H}}| - 1 + \varepsilon$ (initially we set $\varepsilon = 0.5$), and we call 'potential customers' to the remaining $\{c_{f+1}, c_{f+2}, \dots, c_q\}$. We check if $\nu(F^{\mathbb{H}})$ is greater than one and, therefore, the corresponding inequality (38) is violated. Otherwise, for each potential customer $\bar{c} \in \{c_{f+1}, c_{f+2}, \dots, c_q\}$, we iteratively consider the set $\overline{F}^{\mathbb{H}} = F^{\mathbb{H}} \cup \{\bar{c}\}$ and check if $\nu(\overline{F}^{\mathbb{H}})$ is greater than one. Finally, if no violated inequality has been found for any set $\overline{F}^{\mathbb{H}}$, we set $\varepsilon = 0$ and define the set $F^{\mathbb{H}}$ as above (f is now the maximal number such that $z_c^k(F^{\mathbb{H}}) > |F^{\mathbb{H}}| - 1$) and check if $\nu(F^{\mathbb{H}})$ is greater than one. If a subset $F^{\mathbb{H}}$ (or $\overline{F}^{\mathbb{H}}$) for which the corresponding inequality (38) is violated is found, this inequality is added. Then, we look for the cutset of minimum weight between the depot and the arcs of the customer in F_H and the corresponding max-distance inequalities (36) and (37) are checked for violation.

Given a set of customers $F^{\mathbb{H}}$, the value of $\nu(F^{\mathbb{H}})$ (either the number of vehicles needed to service $F^{\mathbb{H}}$ or a lower bound) is computed by solving the corresponding CEARP using the branch-and-cut algorithm in Ávila et al. (2016). Since solving the CEARP instances to optimality can be time consuming, we have limited the execution time of the CEARP solver to 10 seconds.

4.1.5. Inequalities and separation procedures: a summary

Table 1 summarizes the separation procedures described before and provides information on their computational complexity and the references where they were proposed or used. In particular, the first column shows the inequalities class, while column two gives the exact family of inequalities being separated. Column "Separ. Proc." provides the name of the separation procedure used and column "Type" indicates if the procedure is heuristic ("H") or almost exact ("AE"). This last type means that the algorithm is an adaptation of an exact procedure for identifying a violated inequality with similar characteristics. For example, algorithm $A4$ is based on the exact separation of inequalities

$$x^k(\delta^+(S)) \geq \sum_{(i,j)\in H_c} y_{ij}^{kc}, \quad \forall S \subset V\setminus\{1\}, \ \forall c \in \mathbb{H}, \ \forall k \in \mathbb{K}.$$

However, note that the sum in this inequality is done for all the arcs $(i, j) \in H_c$, while in inequalities (13) the sum is for all $(i, j) \in H_c \setminus A(V\setminus S)$. The computational complexity of the algorithms is given in Column 5. An asterisk (*) means that the reported value is the computational complexity of the corresponding separation algorithm if no call to the B&C algorithm described in Ávila et al. (2016) is done. If the B&C is executed to compute the minimum number of vehicles needed to service a given subset of customers, the resulting computational effort is non-polynomial (but each call is limited to 10 seconds). The last two columns indicate if the procedure has already been used in other works and report the corresponding references. A "No/Yes" entry indicates that some new parts have been added in this work to the existing procedures and "[TP]" is used to refer to this paper.

4.2. Comparison of separation strategies and cutting-plane algorithms

To analyze the contribution of the valid inequalities and the separation algorithms presented in the previous sections, we compare the gaps in the root node and the performance profiles (Dolan & More, 2002) of the different versions of our branch-and-cut procedure using different combinations of separation algorithms.

Let $\mathcal{S}$ be the set of versions of our algorithm and $\mathcal{P}$ the set of instances selected for this comparison. Then, for each version $s \in \mathcal{S}$, we calculate $Gap0_s = \frac{1}{|\mathcal{P}|}\sum_{p\in\mathcal{P}}(BKS_p - LB0_{p,s})/LB0_{p,s} * 100$, where BKS_p denotes the value of the optimal or best known solution obtained by any version for instance p and $LB0_{p,s}$ the lower bound obtained by s at the root node. We also compute the *performance ratio* $r_{p,s} = t_{p,s}/\min\{t_{p,s} : s \in \mathcal{S}\}$, where $t_{p,s}$ is the computing time required by algorithm s to solve instance p. If algorithm s is not able to solve the instance p within the time limit, we set $r_{p,s} = \infty$. Thus, the *performance profile* of each version s,

$$\rho_s(\tau) = \frac{|\{p \in \mathcal{P} : r_{p,s} \leq \tau\}|}{|\mathcal{P}|},$$

describes the percentage of instances that can be solved by s within a factor $\tau \geq 1$ compared to the fastest algorithm. Note, for example, that $\rho_s(1)$ is the percentage of instances for which algorithm s is the fastest and that $\rho_s(\infty)$ is the percentage of instances that are solved by algorithm s within the time limit.

We started with a "full version" of the branch and cut, denoted by $V1234$, with the following characteristics. The initial LP relaxation contains all the inequalities in the formulation, except for the connectivity inequalities (6) that are exponential in number and, hence, only the following subset of them are included:

$$x^k(\delta^-(S_c)) \geq z_c^k, \quad \forall k \in \mathbb{K}, \ \forall c \in \mathbb{H},$$

where S_c is the set of vertices incident with the arcs in H_c.

Furthermore, the symmetry breaking inequalities (42)–(44), some max-distance inequalities (38) and (39) associated with some subsets of customers that cannot be serviced with a single vehicle, and inequalities $x^k(\delta^+(1)) \geq 1, \ \forall k \in \mathbb{K}$, which force each vehicle

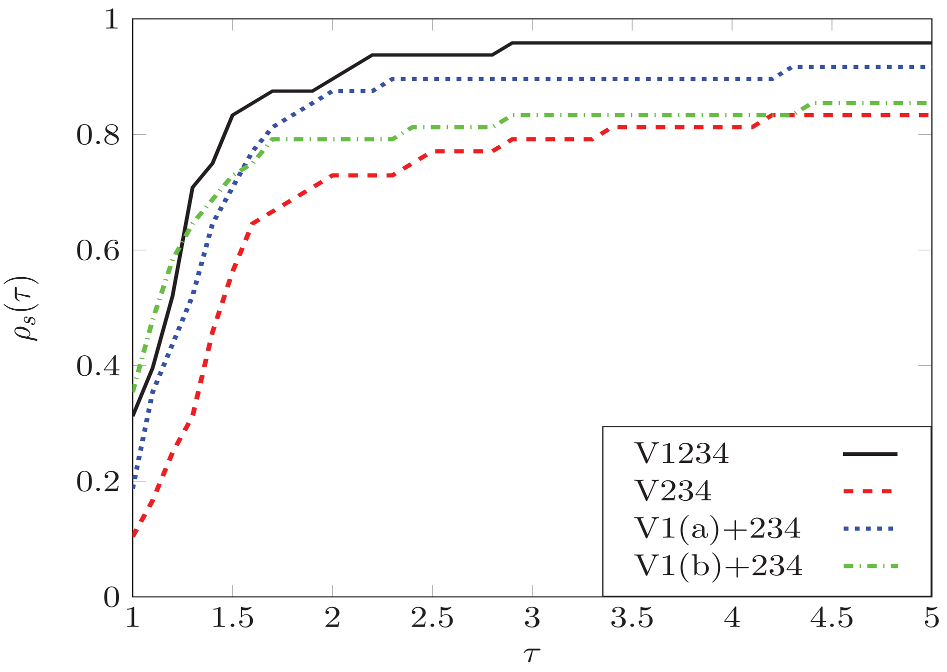


Fig. 7. Impact of the connectivity inequalities: performance profile.

to leave the depot, are included. At each iteration of the cutting-plane algorithm in the root node, the separation procedures described above are applied using the following general scheme and the violated inequalities found are added to the LP relaxation:

1. All the heuristic separation algorithms for connectivity inequalities (A1–A4) are applied. The algorithm based on flow computations (A4) is used only if the other ones fail to find violated inequalities.
2. Heuristic parity separation algorithms (A5–A7).
3. Algorithms for separating K-C, K-C_{02}, and Path-Bridge inequalities (A8–A10) are applied for each vehicle k only if no violated connectivity inequalities have been found for this vehicle.
4. Heuristic algorithms A11 and A12 for separating max-distance inequalities.

Only the fastest separation algorithm for disaggregate connectivity inequalities (A3) is applied in the nodes of the branch-and-cut tree.

The above cutting-plane procedure is applied until no new violated inequalities are found. When this happens, we branch using the Strong Branching strategy implemented in CPLEX with higher priority given to the z_c^k and y_{ij}^{kc} variables.

All other B&C versions are based on different cutting-plane algorithms associated with different separation strategies. The B&C algorithms were tested on a subset of 48 instances taken from the four sets of DC-CEARP instances proposed in Ávila et al. (2017) and whose characteristics are described in Section 5.1. Twelve instances, three for each value of $k \in \{2, 3, 4, 5\}$, were chosen at random from each subset. The experiments were performed on a desktop PC with an Intel(R) Core(TM) i7 at 3.4 gigahertz CPU with 32 gigabyte RAM running Windows 10 Enterprise 64 bits using a single thread. The algorithms were coded in C++ combined with CPLEX 12.10 and all the experiments were carried out with a time limit of 7200 seconds.

We first studied the impact of connectivity inequalities and their separation procedures. To do this, we compared the V1234 B&C with three new versions. In the first version, called V234, we remove all the separation algorithms for connectivity inequalities (A1–A4), except for algorithm A1 when the obtained solution is integer. The second version, V1(a)+234, uses all the separation algorithms except for A2, while the last one, V1(b)+234, uses all the separation algorithms except for A4. Note that A2 and A4 are the most time-consuming procedures.

Table 2
Results on the subset of 48 instances – connectivity.

	# opt	Gap0 (%)	Time0 (seconds)	Time (seconds)
$V1234$	46	5.874	252.38	1031.19
$V234$	41	11.560	198.83	1654.31
$V1(a)+234$	44	5.824	265.86	1209.48
$V1(b)+234$	41	8.693	192.62	1491.66

Fig. 7 shows the performance profile of the four compared versions and Table 2 reports for each version the number of optima obtained (out of 48 instances), the average gap in the root node, and the average computing time spent at the root node and the average total computing time in seconds. V1234, as expected, and surprisingly V1(a)+234, are the best versions in terms of gap, although this last version shows a worse performance profile and worse behavior in terms of averages and number of optima. The other two versions, V234 and V1(b)+234, are clearly dominated by $V1234$. Therefore, we decided to include all separation procedures for connectivity in the final version of the B&C.

Then, we studied the effect of parity inequalities and their separation by comparing the full version V1234 with two versions obtained from it by removing all the separation algorithms for parity inequalities (A5–A7), version V134, and removing algorithms A5 and A7 (V1+2(a)+34). The results obtained are summarized in Fig. 8 and Table 3.

As Fig. 8 shows, all three versions compared have similar performance profiles. However, the gaps in the root node and other measures reported in Table 3 for V134 and V1+2(a)+34 are worse than those of the full version, and therefore none of the latter versions is considered interesting.

We also considered different options regarding the max-distance inequalities. Here, three new versions were implemented.

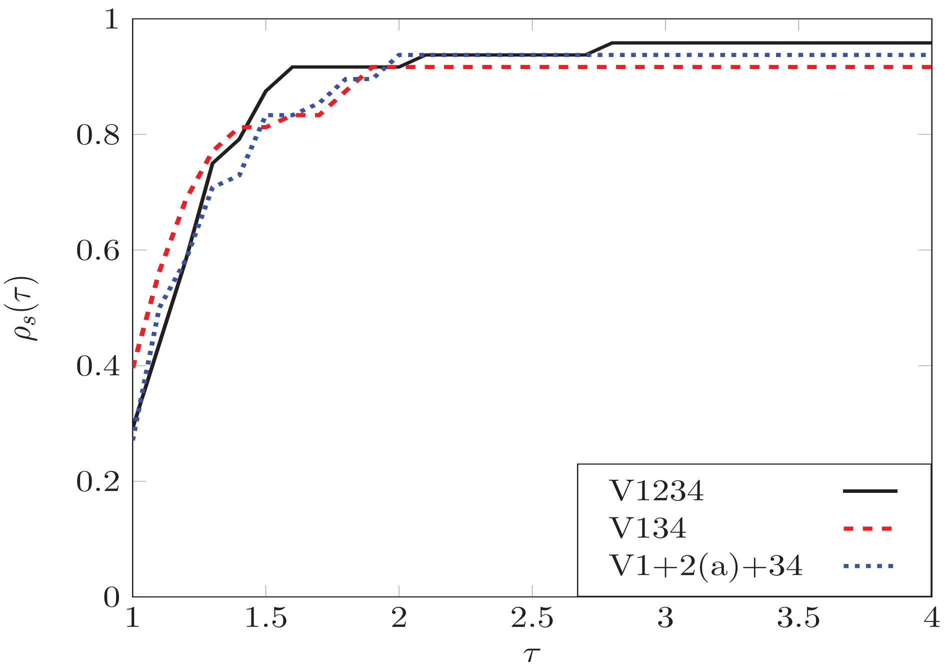


Fig. 8. Impact of the parity inequalities: performance profile.

Table 3
Results on the subset of 48 instances – parity.

	# opt	Gap0 (%)	Time0 (seconds)	Time (seconds)
$V1234$	46	5.874	252.38	1031.19
$V134$	44	6.065	258.00	1069.87
$V1+2(a)+34$	45	6.034	268.13	1165.09

Table 5
Results on the subset of 48 instances – $K-C$, $K-C_{02}$, and Path-Bridge.

	# opt	Gap0 (%)	Time0 (seconds)	Time (seconds)
$V1234$	46	5.874	252.38	1031.19
$V123$	46	9.835	114.34	902.39
$V124$	46	6.064	228.98	1127.72

Table 4
Results on the subset of 48 instances – max-distance.

	# opt	Gap0 (%)	Time0 (seconds)	Time (seconds)
$V1234$	46	5.874	252.38	1031.19
$V123$	46	9.835	114.34	902.39
$V123+4(a)$	46	7.738	136.99	1036.11
$V123+4(b)$	46	6.655	241.18	1054.21

In the first one, V123+4(a), we removed the separation algorithm A11, while algorithm A12 was removed in the second one V123+4(b). The third version, V123, did not include any separation algorithm for max-distance inequalities. These three versions are compared again with version V1234. Performance profiles, average gaps in the root node, and other measures are presented in Fig. 9 and Table 4.

From Fig. 9 we can see that V123 is the fastest version in 60% of the instances, followed by V123+4(a), although they are the two versions with the worst gaps. V123+4(b) is not an interesting option because its performance profile is similar to that of V1234 but it shows worse gaps. The V123+4(a) version has a better performance profile than the full version but a worse average gap, and it is clearly dominated in terms of computing time and performance profile by V123. Therefore, we selected V123 as the most interesting option among the three tested versions.

Finally, we compared a new version V124 resulting from removing the separation algorithms A8-A10 for K-C, K-C$_{02}$, and Path-Bridge inequalities, with the most promising versions obtained from the previous experiments, V1234 and V123. Note that separation algorithms A8, A9, and A10 have many parts in common, and thus removing only some of them would produce no benefit in terms of the overall algorithm efficiency.

Fig. 10 shows the performance profiles for the three versions. Version V123 is the fastest one to reach the optimal solution in more than 80% of the instances and can optimally solve all 46 instances in less that 2 times the time of the fastest version. The performance profile of the other two versions is similar and it can be seen that they reach the 46 optimal solutions only with factors $\tau = 20$ and $\tau = 21$. As for the average gaps in the root node, V124 does not improve the results obtained by V1234 either. Looking at the Table 5, we can see that V124 has no advantage over V1234 and is therefore not considered an interesting alternative. Overall, version V123, although produces greater gaps at the root node, is considerably faster and has a better performance profile than V1234. Therefore, we decided to use version V123 for our final computational experiments.

5. Computational experience

In this section we study the performance of the final version (V123) of the branch-and-cut algorithm, *Algorithm 1* in what follows. As in the previous analysis, the experiments were performed on a desktop PC with an Intel(R) Core(TM) i7 at 3.4 gigahertz CPU with 32 gigabyte RAM running Windows 10 Enterprise 64 bits. Again, we used CPLEX 12.10 with a single thread. The performance of *Algorithm 1* has been compared with that of the best of four branch-and-cut procedures described in Ávila et al. (2017), *Algorithm 2* in what follows, and, for illustrative purposes, with the

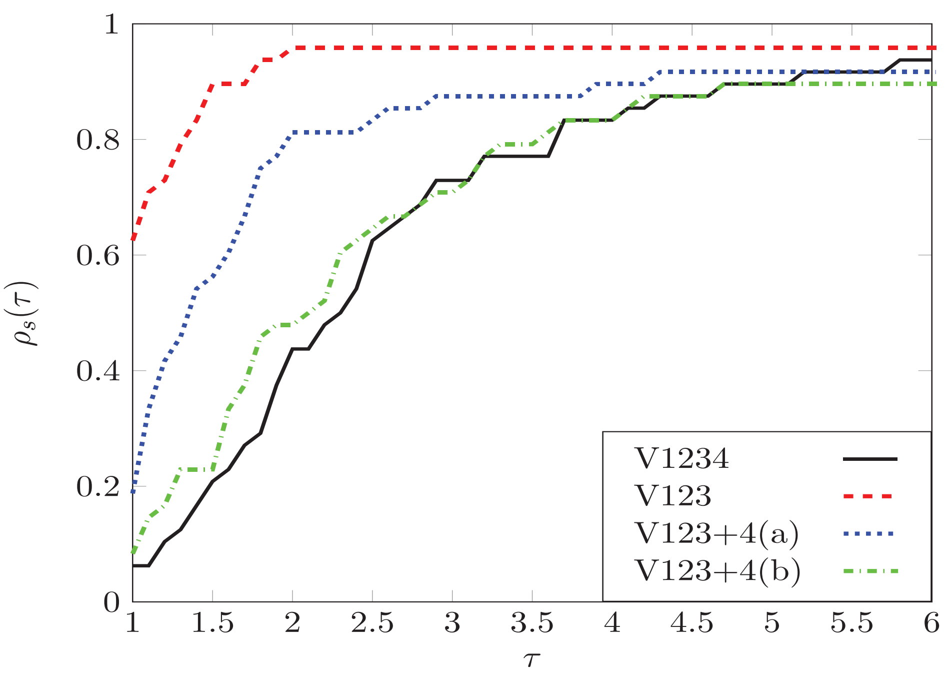


Fig. 9. Impact of the max-distance inequalities: performance profile.

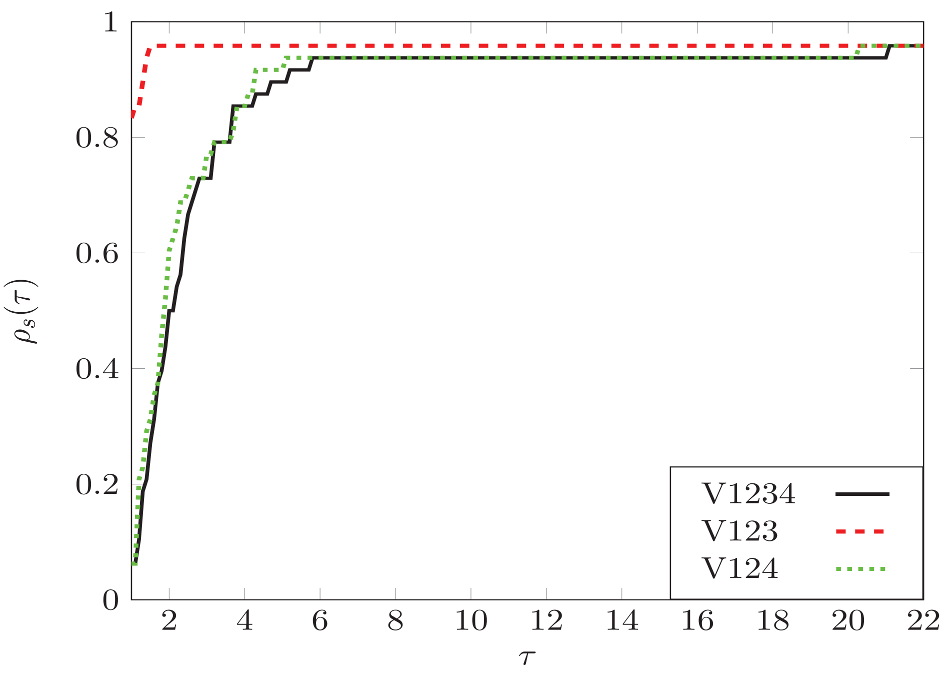


Fig. 10. Impact of the K-C, K-C$_{02}$ and Path-Bridge inequalities: performance profile.

full version V1234, denoted as *Algorithm 0*. *Algorithm 2* has been executed on the same machine and using the same version of CPLEX.

All the experiments were carried out with a time limit of two hours. CPLEX heuristic algorithms were turned off, and CPLEX own cuts, including zero-half cuts, were activated in automatic mode. The optimality gap tolerance was set to zero, best bound strategy was selected and CPLEX presolve phase was reapplied at the end of the root node. The instances used and the computational results obtained are described in what follows.

5.1. Instances

We have tested our branch-and-cut algorithm on the four sets of DC-CEARP instances proposed in Ávila et al. (2017). The graphs of the two first sets of instances, *Random50* and *Random75*, were generated randomly and have 50 and 75 vertices respectively. Sets *Albaida* and *Madrigueras* are based on the street networks of these two Spanish towns. As pointed out in Ávila et al. (2017), generating the value of D_{max} for each instance is a hard task, because depending on this value, the instance can

Table 6
Characteristics of the instances.

	$\|V\|$	$\|A\|$		$\|A_R\|$		$\|A_{NR}\|$		$\|\mathbb{H}\|$	
		Min	Max	Min	Max	Min	Max	Min	Max
Random50	50	296	300	105	292	7	193	10	97
Random75	75	448	450	143	438	10	305	15	140
Albaida	116	259	305	124	172	109	162	18	33
Madrigueras	196	453	544	224	305	197	281	22	47

Table 8
Results on the 11 instances not solved by any algorithm.

	Gap0(%)	Final Gap(%)
Algorithm 0	14.20	8.98
Algorithm 1	14.36	7.80
Algorithm 2	15.35	9.23

Table 9
Results on the 27 instances not solved by at least one algorithm.

	# opt	Gap(%)	Final Gap(%)
Algorithm 0	7	12.07	4.98
Algorithm 1	10	11.86	3.68
Algorithm 2	5	12.94	5.27

be infeasible or trivial (some of the vehicles are not needed). A detailed description of how these values have been generated can be found in that paper. The characteristics of these 251 instances are summarized in Table 6. The complete data, including the values of D_{max} and the number of vehicles with the corresponding best solutions found, can be downloaded from http://www.uv.es/corberan/instancias.htm in the class “Distance-Constrained GDRPP”.

5.2. Computational results

The computational results obtained with *Algorithm 0*, *Algorithm 1*, and *Algorithm 2*, are shown in Table 7, where instances have been grouped by number of vehicles and number of customers, which are shown in columns 1 and 2. Column 3 reports the number of instances of each subset. For both algorithms, the columns labeled ‘# opt’, ‘Gap0 (%)’, and ‘Time’ report the number of optimal solutions found, the average gap in the root node, and the average computing time in seconds, respectively. The bold rows at each group of instances with the same number of vehicles show the total number of instances, in the case of the ‘inst’ and ‘# opt’ columns, and the average values for the remaining columns. The last row of the table summarizes the results for all the instances.

As can be seen in Table 7, the number of optima obtained with *Algorithm 1* is very good (234 out of 251 instances) and is a bit better than those obtained with *Algorithm 0* (231 optima) and *Algorithm 2* (229 optima), although the average gap in the root node is slightly higher than that of *Algorithm 2* (8.94% versus 8.42%), and, as expected, higher than the one obtained using all the separation procedures described in Section 4.1 (6.5%). However, it is in the computing times where we can appreciate the main differences. The average computing time is 976.3 seconds with *Algorithm 1* versus 1080.1 seconds obtained with *Algorithm 2* and 1300.9 of *Algorithm 0*. Although for 2 and 3 vehicles the times for *Algorithm 2* are better on average, when the number of vehicles increases, the times of *Algorithm 1* are significantly lower. The same computational results organized by sets of instances are shown in Tables 10–13 in the Appendix A.4.

To study the running times in more detail, we compare the performance profiles of the three branch-and-cut algorithms (see Fig. 11). Comparing the performance ratios of the three methods at $\tau = 1$, we observe that *Algorithm 1* is the fastest one in almost 80% of the instances, while *Algorithm 0* is the slowest one. As τ increases, the difference between *Algorithm 1* and *Algorithm 2* decreases, but note that five more instances can be solved using *Algorithm 1*.

Table 7 does not report the average final gaps since most instances are optimally solved by all algorithms. Instead, Table 8 compares the average gaps in the root node and the average final gaps obtained by the three procedures in the 11 instances that are not solved by any of the algorithms, while Table 9 provides the same information but in the 27 instances that have not been optimally solved by at least one of the algorithms. Note that

Table 7
Computational results for all the instances grouped by number of vehicles and customers.

Veh	$\|\mathbb{H}\|$	inst	*Algorithm 2* (Ávila et al., 2017)			*Algorithm 1*			*Algorithm 0*		
			# opt	Gap0(%)	Time	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time
2	[10,21]	18	18	1.80	19.2	18	2.4	8.8	18	0.74	36.5
	[12,30]	21	21	1.62	111.6	21	2.6	75.2	21	1.19	240.5
	[31,46]	16	16	1.19	109.3	16	1.9	86.9	16	1.12	256.5
	[47,140]	17	17	1.64	365.1	16	1.6	712.5	16	1.74	1030.8
		72	**72**	**1.57**	**147.8**	**71**	**2.14**	**211.7**	**71**	**1.19**	**379.7**
3	[10,21]	17	17	7.75	40.4	17	8.5	21.0	17	4.67	55.5
	[12,30]	21	21	7.07	238.7	21	7.7	140.4	21	5.70	324.9
	[31,46]	16	16	4.74	696.8	16	5.3	578.7	16	4.32	775.0
	[47,140]	17	16	5.23	1218.9	16	4.6	1477.1	16	5.17	1718.2
		71	**70**	**6.27**	**529.2**	**70**	**6.61**	**530.7**	**70**	**5.02**	**695.5**
4	[10,21]	12	12	14.53	49.6	12	15.8	27.3	12	6.88	71.9
	[12,30]	19	19	14.33	766.9	19	15.6	344.2	19	11.20	1236.5
	[31,46]	16	13	11.65	2253.8	16	11.0	908.1	14	10.05	1790.2
	[47,140]	17	11	9.45	3178.5	13	8.6	3454.6	13	9.02	3862.8
		64	**55**	**12.40**	**1644.7**	**60**	**12.66**	**1251.9**	**58**	**9.52**	**1854.2**
5	[10,21]	9	9	19.35	73.9	9	24.0	60.6	9	6.39	100.7
	[12,30]	10	10	19.63	956.8	10	22.5	589.0	10	17.04	983.1
	[31,46]	10	5	15.64	3963.6	8	15.2	2858.8	7	14.01	3684.5
	[47,140]	15	8	15.61	4516.9	6	14.5	5132.1	6	14.13	5571.8
		44	**32**	**17.29**	**2673.3**	**33**	**18.45**	**2545.6**	**32**	**13.18**	**2980.9**
Total		**251**	**229**	**8.42**	**1080.1**	**234**	**8.94**	**976.3**	**231**	**6.50**	**1300.9**

Table 10
Results for the Random50 instances.

			Algorithm 2 (Ávila et al., 2017)			*Algorithm 1*			*Algorithm 0*		
Veh	$\lvert\mathbb{H}\rvert$	inst	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time
2	10	3	3	1.286	4.15	3	1.654	2.42	3	0.000	4.05
	[24,25]	3	3	0.485	12.96	3	0.064	8.82	3	0.203	23.32
	[45,50]	3	3	0.641	42.42	3	1.717	33.85	3	0.877	90.87
	[92,97]	3	3	2.317	124.04	3	2.242	282.22	3	2.382	333.59
		12	**12**	**1.182**	**45.89**	**12**	**1.419**	**81.83**	**12**	**0.865**	**112.96**
3	10	2	2	4.894	4.58	2	4.950	3.53	2	0.832	5.43
	[24,25]	3	3	2.945	17.06	3	7.099	14.34	3	2.572	25.19
	[45,50]	3	3	4.135	41.79	3	4.167	47.30	3	3.958	102.06
	[92,97]	3	3	7.362	204.18	3	6.629	546.30	3	7.448	665.87
		11	**11**	**4.829**	**72.57**	**11**	**5.780**	**166.45**	**11**	**3.963**	**217.29**
4	[24,25]	3	3	10.132	21.05	3	11.284	19.08	3	8.870	43.11
	[45,50]	3	3	11.697	61.31	3	10.745	91.27	3	9.593	225.55
	[92,97]	3	3	9.457	296.30	3	8.818	897.77	3	9.405	1286.17
		9	**9**	**10.429**	**126.22**	**9**	**10.282**	**336.04**	**9**	**9.290**	**518.28**
5	[45,50]	1	1	23.557	177.30	1	23.713	362.52	1	16.785	247.67
	[92,97]	2	2	15.064	1257.17	1	14.388	3838.74	2	12.212	3652.98
		3	**3**	**17.895**	**897.21**	**2**	**17.496**	**2680.00**	**3**	**13.736**	**2517.87**
Total		**35**	**35**	**6.138**	**147.90**	**34**	**6.447**	**396.49**	**35**	**5.108**	**456.11**

Table 11
Results for the Random75 instances.

			Algorithm 2 (Ávila et al., 2017)			*Algorithm 1*			*Algorithm 0*		
Veh	$\lvert\mathbb{H}\rvert$	inst	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time
2	15	3	3	0.000	8.65	3	0.428	5.91	3	0.000	15.18
	[36,37]	3	3	1.341	38.57	3	1.657	25.69	3	1.022	74.42
	[70,75]	3	3	1.754	117.15	3	1.397	165.33	3	1.796	568.20
	[138,140]	3	3	2.417	1301.58	2	1.927	3224.57	2	2.482	3727.94
		12	**12**	**1.378**	**366.49**	**11**	**1.352**	**855.38**	**11**	**1.325**	**1096.44**
3	15	3	3	9.063	17.45	3	10.801	11.70	3	3.483	29.04
	[36,37]	3	3	4.843	46.26	3	6.286	56.64	3	3.970	97.40
	[70,75]	3	3	6.122	251.25	3	5.843	331.01	3	6.231	1057.64
	[138,140]	3	3	4.516	1469.63	2	3.340	3927.21	2	4.207	4156.80
		12	**12**	**6.136**	**446.15**	**11**	**6.567**	**1081.64**	**11**	**4.473**	**1335.22**
4	15	3	3	11.781	31.76	3	16.834	24.41	3	4.454	68.39
	[36,37]	3	3	15.312	158.04	3	16.322	205.91	3	12.860	371.22
	[70,75]	3	3	10.046	1392.74	3	9.670	2157.84	3	9.146	3341.02
	[138,140]	3	1	6.741	5481.51	1	6.267	6669.73	2	6.645	5945.34
		12	**10**	**10.970**	**1766.01**	**10**	**12.273**	**2264.47**	**11**	**8.276**	**2431.49**
5	15	1	1	13.345	66.69	1	26.376	55.98	1	7.505	102.25
	[36,37]	1	1	32.737	542.60	1	30.940	427.73	1	23.325	855.11
	[70,75]	3	3	14.646	2265.34	2	13.820	4881.76	1	13.076	5785.79
	[138,140]	3	0	16.209	7200.00	0	15.359	7200.00	0	15.960	7200.00
		8	**5**	**17.331**	**3625.66**	**4**	**18.106**	**4591.13**	**3**	**14.742**	**4989.34**
Total		**44**	**39**	**8.192**	**1362.48**	**36**	**8.799**	**1980.61**	**36**	**6.519**	**2233.47**

Table 12
Results for the Albaida instances.

			Algorithm 2 (Ávila et al., 2017)			*Algorithm 1*			*Algorithm 0*		
Veh	$\lvert\mathbb{H}\rvert$	inst	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time
2	18	6	6	3.538	34.25	6	4.034	15.20	6	1.763	58.01
	21	6	6	1.219	16.83	6	2.015	7.12	6	0.457	41.95
	28	6	6	1.970	32.24	6	2.641	18.76	6	1.385	152.69
	33	6	6	1.154	36.06	6	2.575	47.90	6	1.243	114.10
		24	**24**	**1.970**	**29.85**	**24**	**2.816**	**22.24**	**24**	**1.212**	**91.69**
3	18	6	6	8.455	61.72	6	8.943	25.13	6	5.488	73.85
	21	6	6	7.351	42.59	6	8.022	27.41	6	5.735	67.13
	28	6	6	2.997	90.50	6	2.514	44.05	6	2.462	145.30
	33	6	6	4.751	170.06	6	4.609	78.64	6	4.126	270.38
		24	**24**	**5.889**	**91.22**	**24**	**6.022**	**43.81**	**24**	**4.453**	**139.17**
4	18	4	4	12.813	73.04	4	13.200	30.87	4	9.794	78.07
	21	5	5	17.543	41.55	5	17.172	26.10	5	5.995	68.98
	28	6	6	12.621	336.71	6	13.281	177.52	6	12.903	1790.16
	33	6	6	11.328	307.19	6	10.677	135.02	6	10.536	861.09
		21	**21**	**13.460**	**207.78**	**21**	**13.448**	**101.39**	**21**	**9.990**	**788.79**
5	18	3	3	12.669	85.19	3	18.844	47.46	3	4.144	123.25
	21	5	5	24.553	68.58	5	26.648	69.39	5	7.522	86.78
	28	6	6	18.300	557.79	6	23.032	357.12	6	16.521	647.22
	33	3	3	15.245	411.98	3	14.827	404.20	3	13.235	398.89
		17	**17**	**18.606**	**304.77**	**17**	**21.908**	**226.16**	**17**	**11.110**	**346.09**
Total		**86**	**86**	**9.158**	**144.77**	**86**	**10.081**	**87.90**	**86**	**6.216**	**325.45**

Table 13
Results for the Madrigueras instances.

			Algorithm 2 (Ávila et al., 2017)			*Algorithm 1*			*Algorithm 0*		
Veh	$\lvert\mathbb{H}\rvert$	inst	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time	# opt	Gap0(%)	Time
2	22	6	6	1.546	130.35	6	2.930	93.79	6	0.678	253.73
	28	6	6	1.900	221.49	6	3.508	146.33	6	2.011	423.49
	42	6	6	1.229	229.38	6	1.288	166.10	6	1.097	519.50
	47	6	6	1.204	248.47	6	1.161	170.64	6	1.280	573.65
		24	**24**	**1.470**	**207.42**	**24**	**2.222**	**144.22**	**24**	**1.267**	**442.59**
3	22	6	6	10.609	337.95	6	11.965	182.74	6	8.103	421.24
	28	6	6	9.670	398.32	6	8.916	257.54	6	8.103	558.15
	42	6	6	4.608	1659.23	6	5.422	1429.00	6	4.552	1736.76
	47	6	5	4.600	2476.03	6	4.044	1766.47	6	4.588	1887.95
		24	**23**	**7.372**	**1217.88**	**24**	**7.587**	**908.94**	**24**	**6.337**	**1151.03**
4	22	5	5	22.932	1389.99	5	26.800	492.66	5	13.243	1467.63
	28	5	5	10.310	1107.49	5	9.979	590.93	5	8.524	1057.10
	42	6	3	9.993	5616.06	6	8.653	2167.21	4	8.505	3709.02
	47	6	2	9.893	5397.74	4	8.655	4896.17	3	9.493	5563.51
		22	**15**	**12.978**	**3571.37**	**20**	**13.079**	**2172.65**	**17**	**9.855**	**3102.67**
5	22	2	2	21.667	1216.51	2	22.989	1139.78	2	20.618	1974.83
	28	2	2	21.565	1894.25	2	20.549	733.89	2	15.022	998.91
	42	6	1	12.981	6309.60	4	12.805	4491.22	3	12.841	5798.89
	47	6	2	14.655	6110.96	2	12.979	5449.31	2	13.931	6177.68
		16	**7**	**15.768**	**5046.55**	**10**	**15.111**	**3961.91**	**9**	**14.494**	**4862.93**
Total		**86**	**69**	**8.721**	**2250.26**	**78**	**8.894**	**1586.80**	**74**	**7.340**	**2143.17**

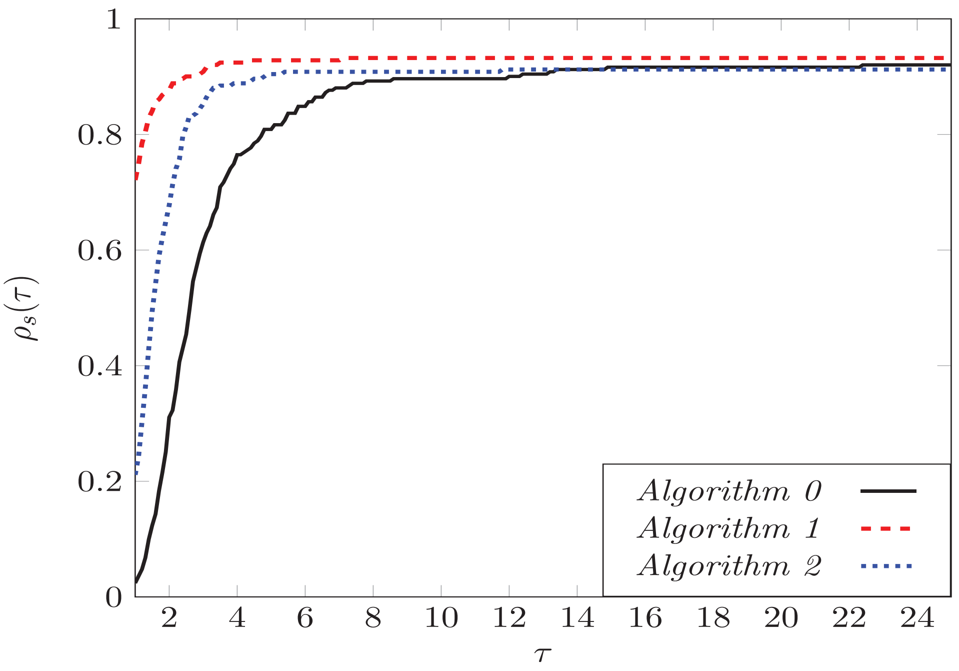


Fig. 11. Performance profile.

in these harder instances, *Algorithm 1* obtains better gaps at the root node, as well as final gaps. Table 9 also reports the number of instances solved optimally by each algorithm.

Looking at the computational results disaggregated by sets of instances, Tables 10–13 in the Appendix A.4, we note that the performance of *Algorithm 1* is better than that of *Algorithm 2* in those instances that are based on real street networks like the Albaida and Madrigueras sets. Another particularity of these two sets of instances is that their number of customers is not too large, from 19 to 34 and from 23 to 48 in the Albaida and Madrigueras instances, respectively, because they are defined following “geographical” criteria, as it was assumed that it can occur in real-life problems. *Algorithm 2*, on the other hand, performs better on the Random 50 and 75 sets, which were randomly generated and have a larger number of customers (many of them defined by larger subsets of arcs).

6. Conclusions

In this paper we study the Distance-Constrained Close Enough Arc Routing Problem, which generalizes the Close Enough Arc Routing Problem to the case in which there is a fleet of vehicles based on a depot that jointly serve a set of customers. Each customer is associated with a set of arcs which are close enough to it, such that the customer can be served by traversing any of these arcs. The length of the routes is limited by a given value and the objective is to minimize the sum of the route distances. The DC-CEARP is inspired by and has application to meter reading problems.

In this work, we propose a new formulation for the DC-CEARP and study its associated polyhedron. Several families of valid inequalities are introduced and separation procedures are devised for them. Extensive computational experiments are carried out to measure the contribution of each of these separation procedures. A branch-and-cut algorithm is presented, that is able to solve to optimality instances with up to 140 customers, 196 vertices, 544 arcs, and 5 vehicles.

In what refers to future work, and taking into account the kind of applications of this problem, we plan to study the variant in which the routes of the vehicles have to be balanced. One way of balancing routes consists of minimizing the length of the longest one. For this problem we are planning to design and implement a branch-and-cut-and-price algorithm capable of solving instances with a large number of vehicles.

Acknowledgments

This work was supported by the Spanish Ministerio de Ciencia, Innovación y Universidades (MICIU) and Fondo Social Europeo (FSE) through project PGC2018-099428-B-I00. The authors want to thank the comments and suggestions done by two anonymous referees that have contributed to improve the content and readability of the article.

A.1. Proof of Theorem 1

Theorem 1. *For each vehicle k, disaggregate K-C inequalities (20) and (21) are valid for the DC-CEARP.*

Proof. Let us suppose that $1 \in M_0 \cup M_Q$ (the proof for the case $1 \notin M_0 \cup M_Q$ is similar). We have to prove that all the routes $(\bar{x}^k, \bar{y}^k)$ for vehicle k corresponding to DC-CEARP solutions satisfy inequality (20). We consider the following cases:

(a) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, $i = 1, \dots, q$, and servicing each customer c_j from a required arc in G_j, $j = 1, \dots, Q-1$. On the one hand, these tours $\bar{x}^k$ traverse at least q times the arcs in (M_0, M_Q), and visit at least once each node set $M_0 \cup M_Q$, $M_1, \dots, M_{Q-1}$, and, hence, they satisfy (19):

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}_{ij} \geq (Q-2)q + 2(Q-1).$$

Additionally, variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, q$, and $\bar{y}^{kc_j}(G_j) = 1$, for each $j = 1, \dots, Q-1$. Substituting them in the RHS of (20), we obtain

$$(Q-2)\sum_{i=1}^{q}(2\bar{y}^{kc_i}(F_i) - 1) + \sum_{j=1}^{Q-1} 2\ \bar{y}^{kc_j}(G_j)$$
$$= (Q-2)q + 2(Q-1).$$

Hence, $\sum_{(i,j)\in A}\alpha_{ij}\bar{x}_{ij} \geq (Q-2)\sum_{i=1}^{q}(2\bar{y}^{kc_i}(F_i) - 1) + \sum_{j=1}^{Q-1} 2\ \bar{y}^{kc_j}(G_j)$ holds and routes $(\bar{x}^k, \bar{y}^k)$ satisfy inequality (20).

(b) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, $i = 1, \dots, q$, and each customer c_j from a required arc in G_j, $j = 1, \dots, Q-1$, except one of them, say c_l. These tours $\bar{x}^k$ traverse q required arcs between M_0 and M_Q and visit all but one the subgraphs $G(M_1), \dots, G(M_{Q-1})$. Note that, regarding a K-C structure (see Fig. 2), this cannot be done at an α-cost lower than $(Q-2)q + 2(Q-1) - 2$ (otherwise, by adding two arcs connecting M_l with M_{l-1} we would obtain a tour satisfying (a) and (b) with α-cost less than $(Q-2)q + 2(Q-1)$, which is impossible) and, hence, these tours satisfy

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}_{ij} \geq (Q-2)q + 2(Q-1) - 2.$$

On the other hand, variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, q$, and $\bar{y}^{kc_j}(G_j) = 1$, for all $j = 1, \dots, Q-1$, except one of them, for which $\bar{y}^{kc_l}(G_l) = 0$. Thus, if we substitute these values in the RHS of (20),

$$(Q-2)\sum_{i=1}^{q}(2\bar{y}^{kc_i}(F_i) - 1) + \sum_{j=1}^{Q-1} 2\ \bar{y}^{kc_j}(G_j)$$
$$= (Q-2)q + 2(Q-2) = (Q-2)q + 2(Q-1) - 2,$$

is obtained and, thus, $(\bar{x}^k, \bar{y}^k)$ satisfies (20).

(c) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, $i = 1, \dots, q$, and each customer c_j from a required arc in G_j, $j = 1, \dots, Q-1$, except a number b of them ($b = 2, 3, \dots$). As before, it can be seen that these tours $\bar{x}^k$ satisfy

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}_{ij} \geq (Q-2)q + 2(Q-1) - 2b,$$

and the RHS of inequality (20) takes the value

$$(Q-2)\sum_{i=1}^{q}(2\bar{y}^{kc_i}(F_i) - 1) + \sum_{j=1}^{Q-1} 2\ \bar{y}^{kc_j}(G_j)$$
$$= (Q-2)q + 2(Q-1-b) = (Q-2)q + 2(Q-1) - 2b.$$

(d) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, for all $i = 1, \dots, q$ except one of them, say c_l, and each customer c_j from a required arc in G_j, $j = 1, \dots, Q-1$. These tours $\bar{x}^k$ traverse $q-1$ (an odd number) required arcs between M_0 and M_Q and visit all the subgraphs $G(M_1), \dots, G(M_{Q-1})$. Regarding a K-C structure, this cannot be done at an α-cost lower than $(Q-2)(q-2) + 2(Q-1)$ (otherwise, by adding two arcs connecting M_0 with M_Q, with α-cost $Q-2$ each, we would obtain a tour satisfying (a) and (b) with α-cost less than $(Q-2)q + 2(Q-1)$, which is impossible) and, hence, these tours satisfy

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}_{ij} \geq (Q-2)(q-2) + 2(Q-1).$$

Moreover, variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, q$ except one of them, for which $\bar{y}^{kc_l}(F_l) = 0$, and $\bar{y}^{kc_j}(G_j) = 1$, for all $j = 1, \dots, Q-1$. Thus, after substituting these values in the RHS of (20) we obtain

$$(Q-2)\sum_{i=1}^{q}(2\bar{y}^{kc_i}(F_i) - 1) + \sum_{j=1}^{Q-1} 2\ \bar{y}^{kc_j}(G_j)$$
$$= (Q-2)(q-1-1) + 2(Q-1),$$

and $(\bar{x}^k, \bar{y}^k)$ satisfies (20).

(e) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, for all $i = 1, \dots, q$ except two of them, and each customer c_j from a required arc in G_j, $j = 1, \dots, Q-1$. These tours $\bar{x}^k$ traverse $q-2$ (an even number) required arcs between M_0 and M_Q and visit all the subgraphs $G(M_1), \dots, G(M_{Q-1})$, so they satisfy

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}_{ij} \geq (Q-2)(q-2) + 2(Q-1).$$

Variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, q$ except two of them, for which $\bar{y}^{kc_l}(F_l) = 0$, and $\bar{y}^{kc_j}(G_j) = 1$, for all $j = 1, \dots, Q-1$, and the RHS of inequalities (20) takes the value

$$(Q-2)\sum_{i=1}^{q}(2\bar{y}^{kc_i}(F_i) - 1) + \sum_{j=1}^{Q-1} 2\ \bar{y}^{kc_j}(G_j)$$
$$= (Q-2)(q-2-1-1) + 2(Q-1) < (Q-2)(q-2) + 2(Q-1),$$

and $(\bar{x}^k, \bar{y}^k)$ satisfies (20).

(f) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, for all $i = 1, \dots, q$ except three, four,...of them, and each customer c_j from a required arc in G_j, $j = 1, \dots, Q-1$. By using a similar reasoning, it can be proved that they satisfy inequality (20).

(g) Routes $(\bar{x}^k, \bar{y}^k)$ similar to those in the cases (d)–(f) but where each customer c_j is serviced from a required arc in G_j, $j = 1, \dots, Q-1$, except a number b of them ($b = 1, 2, \dots$). It can be seen that both the term $\sum \alpha_{ij}\bar{x}_{ij}$ and the RHS of inequality (20) decrease in $2b$ units, thus satisfying inequality (20). $\square$

A.2. Proof of Theorem 2

Theorem 2. *Given a set of vehicles $\Omega \subseteq \mathbb{K}$, the Ω-aggregate K-C inequalities (23) and (24) are valid for the DC-CEARP.*

Proof. Again, let us suppose that $1 \in M_0 \cup M_Q$ (the proof for the case $1 \notin M_0 \cup M_Q$ is similar). We have to prove that every DC-CEARP solution satisfies inequality (23). Let $(\bar{x}^1, \bar{y}^1, \dots, \bar{x}^K, \bar{y}^K)$ be a DC-CEARP solution. Then, $\sum_{k\in\Omega} \bar{x}^k$ is a tour on the arcs of G since it represents a connected and even graph. On the other hand, for each $i = 1, \dots, q$, the sum $\sum_{k\in\Omega} \bar{y}^{kc_i}(F_i)$, is a binary value indicating if any of the vehicles in Ω services the customer c_i from an arc in F_i (see inequalities (3)). In the same way, for each $j = 1, \dots, Q-1$, the sum $\sum_{k\in\Omega} \bar{y}^{kc_j}(G_j)$ is a binary value indicating if any of the vehicles in Ω services the customer c_j from an arc in G_j. Hence, a similar reasoning to that of the proof of Theorem 1, but replacing $(\bar{x}^k, \bar{y}^k)$ by $(\sum_{k\in\Omega} \bar{x}^k, \sum_{k\in\Omega} \bar{y}^k)$, concludes that the following inequality, which can be rewritten as inequality (23), is satisfied:

$$\sum_{(i,j)\in A} \alpha_{ij} \sum_{k\in\Omega} \bar{x}^k_{ij} \geq (Q-2) \sum_{i=1}^{q} \left(2 \sum_{k\in\Omega} \bar{y}^{kc_i}(F_i) - 1 \right) + \sum_{j=1}^{Q-1} 2 \sum_{k\in\Omega} \bar{y}^{kc_j}(G_j).$$

$\square$

A.3. Proof of Theorem 5

Theorem 5. *For each vehicle k, disaggregate Path-Bridge inequalities (29) and (30) are valid for the DC-CEARP.*

Proof. Let us suppose that $1 \in M_0 \cup M_Z$ (the proof for the case $1 \notin M_0 \cup M_Z$ is similar). We have to prove that all the single routes $(\bar{x}^k, \bar{y}^k)$ for vehicle $k \in \mathbb{K}$ corresponding to DC-CEARP solutions satisfy inequality (29). We consider the following cases:

(a) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from an arc in F_i, $i = 1, \dots, B$, and servicing each customer c_j from an arc in G^t_r, $t = 1, \dots, P$ and $r = 1, \dots, n_t$. On the one hand, these tours $\bar{x}^k$ traverse at least B times the arcs in (M_0, M_Z), and visit at least once all the node sets $M_0 \cup M_Z$ and M^t_r. It can be seen (see Corberán et al., 2001) that these tours satisfy:

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}^k_{ij} \geq B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1.$$

On the other hand, variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, B$, and $\bar{y}^{kc_j}(G^t_r) = 1$, for each $t = 1, \dots, P$ and $r = 1, \dots, n_t$. Substituting these values in the RHS of (29) we obtain

$$\sum_{i=1}^{B} \left(2\bar{y}^{kc_i}(F_i) - 1 \right) + \sum_{t=1}^{P} \sum_{j=1}^{n_t} \frac{2\bar{y}^{kc_j}(G^t_j)}{n_t - 1} - P + 1 = B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1.$$

Hence, $\sum_{(i,j)\in A} \alpha_{ij}\bar{x}^k_{ij} \geq \sum_{i=1}^{B} \left(2\bar{y}^{kc_i}(F_i) - 1 \right) + \sum_{t=1}^{P} \sum_{j=1}^{n_t} \frac{2\bar{y}^{kc_j}(G^t_j)}{n_t - 1} - P + 1$ holds, and routes $(\bar{x}^k, \bar{y}^k)$ satisfy inequality (29).

(b) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, $i = 1, \dots, B$, and servicing each customer c_j, except one of them ($H_{c_l} \in G^{t_0}_{r_0}$), from a required arc in G^t_r, $t = 1, \dots, P$, $r = 1, \dots, n_t$. These tours $\bar{x}^k$ traverse B times some required arcs between M_0 and M_Z and visit all the sets M^t_r except the set $M^{t_0}_{r_0}$. Note that this cannot be done with an α-cost lower than $B + \sum_{t=1}^{P} \frac{2n_t}{n_t-1} - P + 1 - \frac{2}{n_{t_0}-1}$. Otherwise, by adding two arcs connecting $M^{t_0}_{r_0}$ with $M^{t_0}_{r_0-1}$, with α-cost $\frac{2}{n_{t_0}-1}$, we would obtain a tour, traversing at least B times the arcs in (M_0, M_Z) and visiting all the node sets $M_0 \cup M_Z$ and M^t_r, with α-cost less than $B + \sum_{t=1}^{P} \frac{2n_t}{n_t-1} - P + 1$, which is impossible. Hence, these tours $\bar{x}^k$ satisfy

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}^k_{ij} \geq B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1 - \frac{2}{n_{t_0} - 1}.$$

Moreover, variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, B$, and $\bar{y}^{kc_j}(G^t_r) = 1$, for all $t = 1, \dots, P$ and $r = 1, \dots, n_t$, except one of them, for which $\bar{y}^{kc_l}(G^{t_0}_{r_0}) = 0$. Thus, the RHS of inequalities (29) takes the value

$$(2B - B) + \sum_{\substack{t=1 \\ t \neq t_0}}^{P} \frac{2n_t}{n_t - 1} + \frac{2(n_{t_0} - 1)}{n_{t_0} - 1} - P + 1$$

$$= B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1 - \frac{2}{n_{t_0} - 1}$$

and, hence, the routes $(\bar{x}^k, \bar{y}^k)$ satisfy (29).

(c) Routes $(\bar{x}^k, \bar{y}^k)$ servicing each customer c_i from a required arc in F_i, for all $i = 1, \dots, B$ except one of them, say c_l, and each customer c_j from a required arc in G^t_r, $t = 1, \dots, P$, $r = 1, \dots, n_t$.

Tours $\bar{x}^k$ traverse $B-1$ required arcs between M_0 and M_Z and visit all the sets M^t_r. Considering that $P + B - 1$ is an even number, this cannot be done with a α-cost lower than $B + \sum_{t=1}^{P} \frac{2n_t}{n_t-1} - P + 1 - 2$. Otherwise, by adding two arcs connecting M_0 with M_Z, with α-cost 1 each, we would obtain a tour, traversing at least B times the arcs in (M_0, M_Z) and visiting all the node sets $M_0 \cup M_Z$ and M^t_r, with α-cost less than $B + \sum_{t=1}^{P} \frac{2n_t}{n_t-1} - P + 1$, which is impossible. Hence, these tours $\bar{x}^k$ satisfy

$$\sum_{(i,j)\in A} \alpha_{ij}\bar{x}_{ij} \geq= B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1 - 2.$$

Additionally, variables $\bar{y}^k$ satisfy $\bar{y}^{kc_i}(F_i) = 1$, for each $i = 1, \dots, B$ except one of them, for which $\bar{y}^{kc_l}(F_l) = 0$, and $\bar{y}^{kc_j}(G^t_r) = 1$, for all $t = 1, \dots, P$, $r = 1, \dots, n_t$. Thus, after substituting them in the RHS of (29), we obtain

$$2(B-1) - B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1 = B + \sum_{t=1}^{P} \frac{2n_t}{n_t - 1} - P + 1 - 2,$$

and the routes $(\bar{x}^k, \bar{y}^k)$ satisfy (29).

(d) In a similar way, it can be seen that inequalities (29) are satisfied by all the routes $(\bar{x}^k, \bar{y}^k)$ servicing any other number of customers c_i and c_j. $\square$

A.4. Computational results per set of instances

In this appendix we show, for each set of instances, the computational results obtained with the proposed branch and cut and their comparison with those obtained with the exact algorithm described in Ávila et al. (2017) and with the version using all the separation procedures. Tables 10, 11, 12, and 13 report the results for the sets Random50, Random75, Albaida, and Madrigueras, respectively.

References

Aráoz, J., Fernández, E., & Franquesa, C. (2017). The generalized arc routing problem. *TOP, 25*(3), 497–525.

Ávila, T., Corberán, Á., Plana, I., & Sanchis, J. M. (2016). A new branch-and-cut algorithm for the generalized directed rural postman problem. *Transportation Science, 50*, 750–761.

Ávila, T., Corberán, Á., Plana, I., & Sanchis, J. M. (2017). Formulations and exact algorithms for the distance-constrained generalized directed rural postman problem. *EURO Journal on Computational Optimization, 5*, 339–365.

Behdani, B., & Smith, J. (2014). An integer-programming-based approach to the close-enough traveling salesman problem. *INFORMS Journal on Computing, 26*, 415–432.

Carrabs, F., Cerrone, C., Cerulli, R., & Gaudioso, M. (2017). A novel discretization scheme for the close enough traveling salesman problem. *Computers & Operations Research, 78*, 163–171.

Cerrone, C., Cerulli, R., Golden, B., & Pentangelo, R. (2017). A flow formulation for the close-enough arc routing problem. In S. A., & S. C. (Eds.), *Optimization and decision science: Methodologies and applications, ODS 2017*. In *Springer proceedings in mathematics & statistics: 217* (pp. 539–546). Springer.

Corberán, A., & Laporte, G. (2014). Arc routing: Problems, methods, and applications. *MOS-SIAM series on optimization*. SIAM, Philadelphia.

Corberán, A., Eglese, R., Hasle, G., Plana, I., & Sanchis, J. (2020). Arc routing problems: A review of the past, present, and future. *Networks*. https://doi.org/10.1002/net.21965.

Corberán, A., Letchford, A., & Sanchis, J. (2001). A cutting plane algorithm for the general routing problem. *Mathematical Programming, 90*, 291–316.

Corberán, Á., Plana, I., Reula, M., & Sanchis, J. (2019). A matheuristic for the distance-constrained close-enough arc routing problem. *TOP, 27*, 312–326.

Corberán, A., Romero, A., & Sanchis, J. (2003). The mixed general routing problem polyhedron. *Mathematical Programming, 96*, 103–137.

Corberán, A., & Sanchis, J. (1994). A polyhedral approach to the rural postman problem. *European Journal of Operational Research, 79*, 95–114.

Cornuèjols, G., Fonlupt, J., & Naddef, D. (1985). The traveling salesman problem on a graph and some related inequalities. *Mathematical Programming, 33*, 1–27.

Coutinho, W., Subramanian, A., do Nascimento, R., & Pessoa, A. (2016). A branch-and-bound algorithm for the close enough traveling salesman problem. *INFORMS Journal on Computing, 28*, 752–765.

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

Dong, J., Yang, N., & Chen, M. (2007). Heuristic approaches for a TSP variant: The automatic meter reading shortest tour problem. In *Extending the horizons: Advances in computing, optimization, and decision technologies* (pp. 145–163). Springer.

Drexl, M. (2007). *On some generalized routing problems*. Rheinisch-Westfälische Technische Hochschule, Aachen University Ph.D. thesis..

Drexl, M. (2014). On the generalized directed rural postman problem. *Journal of the Operational Research Society, 65*, 1143–1154.

Eglese, R., Golden, B., & Wasil, E. (2014). Route optimization for meter reading and salt spreading, Philadelphia. In A. Corberán, & G. Laporte (Eds.), *MOS-SIAM Series on Optimization*. *Arc routing: Problems, methods and applications* (pp. 303–320). SIAM, Philadelphia.

Fischetti, M., Salazar-González, J. J., & Toth, P. (1995). Experiments with a multi–commodity formulation for the symmetric capacitated vehicle routing problem. In *Proceedings of the 3rd meeting of the euro working group on transportation* (pp. 169–173).

Gendreau, M., Laporte, G., & Semet, F. (1997). The covering tour problem. *Operations Research, 45*(4), 568–576.

Gulczynski, D., Heath, J., & Price, C. (2006). The close enough traveling salesman problem: A discussion of several heuristics. In *Perspectives in operations research*. In *Operations research/computer science interfaces series: 36* (pp. 217–283). Springer.

Hà, M.-H., Bostel, N., Langevin, A., & Rousseau, L.-M. (2012). An exact algorithm for close enough traveling salesman problem. In *Proceedings of the 1st international conference on operations research and enterprise systems* (pp. 233–238).

Hà, M.-H., Bostel, N., Langevin, A., & Rousseau, L.-M. (2014). Solving the close enough arc routing problem. *Networks, 63*, 107–118.

Letchford, A. (1997). New inequalities for the general routing problem. *European Journal of Operational Research, 96*, 317–322.

Mennell, W. (2009). *Heuristics for solving three routing problems: Close-enough traveling salesman problem, close-enough vehicle routing problem, sequence-dependent team orienteering problem*. College Park: University of Maryland Ph.D. thesis..

Mourão, M. C., & Pinto, L. S. (2017). An updated annotated bibliography on arc routing problems. *Networks, 70*, 144–194.

Renaud, A., Absi, N., & Feillet, D. (2017). The stochastic close-enough arc routing problem. *Networks, 69*, 205–221.

Shuttleworth, R., Golden, B., Smith, S., & Wasil, E. (2008). Advances in meter reading: Heuristic solution of the close enough traveling salesman problem over a street network. In B. Golden, S. Raghavan, & E. Wasil (Eds.), *The vehicle routing problem: Latest advances and new challenges* (pp. 487–501). Springer.

Uribe-Pérez, N., Hernández, L., De la Vega, D., & Angulo, I. (2016). State of the art and trends review of smart metering in electricity grids. *Applied Sciences, 6*(68), 1–24.

Yuan, B., Orlowska, M., & Sadiq, S. (2007). On the optimal robot routing problem in wireless sensor networks. *IEEE Transactions on Knowledge and Data Engineering, 19*(9), 1252–1261.