The Generalized Arc Routing Problem

Julián Aráoz [*1], Elena Fernández [†1], and Carles Franquesa [‡2]

[1]Department d'Estadística i Investigació Operativa, Universitat Politècnica de Catalunya-BcnTech, Spain
[2]Área de Matemáticas y Algoritmia, IKIAM Universidad Regional Amazónica, Ecuador

Abstract

In this paper the Generalized Arc Routing Problem (GARP) is focused. The GARP is stated on an undirected graph in which some clusters are defined as pairwise-disjoint connected subgraphs, and a route is sought that traverses at least one edge of each cluster. Broadly speaking, the GARP is the arc routing counterpart of the Generalized Traveling Salesman Problem (GTSP) [27, 28]. In the GTSP, the set of vertices of a given graph is partitioned into clusters and a route is sought that visits at least one vertex of each cluster. Different versions of the GTSP have been studied by various authors and their properties have been established. It is well-known that the GTSP can be used to model several node and arc routing problems [29], like the TSP and the Undirected Rural Postman Problem (RPP).

1 Introduction

The Generalized Arc Routing Problem is defined on an undirected graph with disjoint clusters of demand edges. The problem itself is to find the minimum cost tour (closed walk) that visits at least one edge out of each cluster. The problem might be seen as the arc routing counterpart of the Generalized Traveling Salesman Problem, that is a well-known node routing problem which extends the Traveling Salesman Problem. In the GTSP the set of vertices of a given graph is partitioned into clusters and a route is sought that visits at least one vertex of each cluster. The GTSP can also be used to model several arc routing problems, like the Rural Postman Problem, [1]. However, arc routing problems have received an increasing interest in the last decades, and highly specialized solution algorithms have been developed in the last years able to solve very efficiently different classes of arc routing problems, see [2, 13].

Potential applications of the GARP arise in different contexts, which can be cast within arc routing. For instance, in meter reading, one of the classical applications of arc routing, current technologies make it possible to read the requested meters by traversing just a few of the edges where meters have to be read instead of all of them. Also in quality control for networks maintenance only a small subset of the edges of a network has to be traversed. Furthermore, the GARP is most appropriate for modeling location/arc-routing problems in which facilities have to be located at some given areas (clusters) and connected among them by means of a route. Depending on their characteristics, facilities cannot be located at the vertices of a network (warehouses, pickup/delivery points, etc.), so they have to be located at different edges of the clusters, and the design of the connecting route involves arc-routing decisions. The overall location/arc-routing problem is thus a GARP.

[*]julian.araoz@upc.edu
[†]e.fernandez@upc.edu
[‡]carles.franquesa@ikiam.edu.ec

Directed versions of a more general problem, in which the clusters do not necessarily have to be disjoint have been studied by several authors under different names, as discussed in [8]. M. Drexl studied the Generalized Directed Rural Postman Problem (GDRPP) in [19, 20], where no depot is assumed, established its NP-hardness and proposed a branch-and-cut algorithm. Previously in [32] the authors introduced the so-called Close-Enough Traveling Salesman Problem and proposed heuristics to solve large instances. The same problem has been studied more recently under the name of Close-Enough Arc Routing Problem in [25, 8], where formulations, valid inequalities and branch-and-cut algorithms have been proposed. A multi-vehicle version of the GDRPP has been studied in [16].

Preliminary results of our work on the GARP have been presented in [5, 6, 22, 23]. To the best of our knowledge, however, the GARP has not been addressed in the literature. In this work we introduce the GARP as a single vehicle arc routing problem on an undirected graph. This allows us to easily derive optimality conditions which guarantee that in any optimal solution no edge will be traversed more than twice. Based on this, we present a first linear integer formulation for the GARP, which uses two sets of binary variables, in the spirit of current formulations for this type of problems. After analyzing some dominance conditions, a tighter formulation with only one set of binary variables is proposed. The polyhedron associated with the latter formulation is studied and some facets and families of valid inequalities are given. In particular, we present two new families of inequalities which are valid for the GARP and extend the well-known co-circuit [9] and matching-type inequalities for odd subsets of vertices introduced by Edmonds [21], and we establish some relationship between them. We also study the separation problem for the different families of valid inequalities, and propose a solution algorithm which iteratively reinforces the current LP relaxation by incorporating separated inequalities. Finally, we report on the numerical results of a series of computational experiments.

The paper is structured as follows. Section 2 defines the problem and presents the first formulation with two sets of binary variables. Section 3 presents the dominance relations, which allow to formulate the GARP using only one set of binary variables and compares the two formulations. In Section 4 the polyhedron associated with the second formulation is studied, stating its dimension, and presenting some families of facets, whereas in Section 5 some families of valid inequalities are presented, including extensions of classical co-circuit and matching inequalities, which are valid for the GARP. The separation of the different families of inequalities is addressed in Section 6. The proposed solution algorithm is presented in Section 7 and the numerical results obtained in the computational experiments are analyzed in Section 8.

2 The Generalized Arc Routing Problem

The GARP is defined on an undirected connected graph $G = (V, E)$ with a distinguished vertex $v_d \in V$, the depot. With each edge $(u, v) \in E$ is associated a non-negative cost, c_{uv}. A set of subgraphs of G is given, $C_k = (V_k, D_k)$, $k \in K$ with $\emptyset \neq V_k \subset V$, $\emptyset \neq D_k \subset E$, $k \in K$, and $V_k \cap V_{k'} = \emptyset$, $k, k' \in K, k \neq k'$. Subgraphs C_k are referred to as *clusters* and edges in $D = \cup_{k \in K} D_k$ as *demand* edges. Note that clusters are not required to be connected. Throughout, we denote $R = E \setminus D$.

We assume no demand edge is incident with the depot. If necessary, a new depot is defined, v'_d, connected to the original one with a zero cost non-demand edge, and with every other vertex $u \in V \setminus \{v_d\}$ with a non-demand edge (v'_d, u) of cost $c_{v_d u}$. We further assume that G has been simplified so that V is the set of vertices incident with edges in D plus the depot, and E contains all edges of D plus additional non-demand edges, connecting every pair of vertices u, v not connected of D, and with $c_{u,v}$ equal to that of the shortest path in the original graph.

Feasible solutions to the GARP are tours (closed walks) starting and ending at the depot, which traverse at least one edge of each cluster. The GARP is to find a minimum cost feasible tour.

Throughout, the following standard notation will be used: For a given subset of vertices $S \subset V$, $E(S)$ means

the subset of edges of E with both end-vertices in S, and for a given subset of edges $F \subset E$, $V(F) \subset V$ is the subset of vertices incident with some edge of F. Also, $\delta(S) = \{e \in E \mid e = (u, v), u \in S, v \in V \setminus S\}$ denotes the cut-set between S and $V \setminus S$, although for a singleton we simply write $\delta(v) = \delta(\{v\})$. Additionally, for any $F \subset E$ we write $E_F = E \cap F$ and $\delta_F(S) = \delta(S) \cap F$. If S_1 and S_2 are two disjoint vertex sets, $(S_1 : S_2)$ represents the set of edges with an end-vertex in S_1 and another end-vertex in S_2. Thus, $\delta(S) = (S : V \setminus S)$. And finally, we use the compact notation $f(A) = \sum_{e \in A} f_e$ where $A \subseteq E$, and f is a vector or function defined on E.

Proposition 2.1 *The GARP is NP-hard.*

Proof: This proof is based on considering the GARP when all clusters have just one edge. Let start from the well-known Rural Postman Problem defined on an undirected graph $G(V, E)$ where there is a proper subset $R \subset E$ that must be traversed in any feasible solution. Edges in R are called the required edges.

A polynomial reduction from the RPP to the GARP is shown in Algorithm 1. Required edges in G are interpreted as demand edges in the reduced instance G'. For brevity, the treatment of edge costs has been omitted. It would simply consist of maintaining original costs for the replacements done into the internal loop, and zeroing those corresponding to the cliques ending the main one.

```
procedure RPP_To_GARP(G)
    G' = G
    for u ∈ V with |δ_R(u)| > 1:
        S = ∅
        for w = (u, v) ∈ δ_R(u):
            u_w = new vertex()
            S = S ∪ {u_w}
            V' = V' ∪ {u_w}
            E' = E' \ {w}
            E' = E' ∪ {(u_w, v)}
        endfor
        E' = E' ∪ clique(S)
    endfor
    return G'
```

Algorithm 1 *Polynomial Reduction from RPP to GARP.*

By using this reduction, an optimal solution for the GARP instance G' would also be an optimal solution for the corresponding RPP instance G where demand edges of the first would be interpreted as requires edges of the second. In Figure 1 this graphical transformation is illustrated.

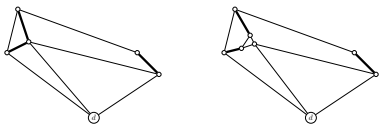

Figure 1: *left: RPP Instance; right: Reduced GARP Instance.*

Therefore, the GARP is NP-hard. ■

Similarly to other single vehicle arc routing problems on undirected graphs with non-negative costs, it is easy to see that, for a given GARP instance, an optimal solution exists in which no edge is traversed more than twice. Otherwise, two copies of the same edge can be removed without affecting neither the condition that at least one demand edge of each cluster is traversed, nor the parity of the vertices or the connectivity with the depot. It is easy to see that, in contrast to the RPP on an undirected graph [24], the edges that can be traversed twice in an optimal GARP solution are not limited to the edges of the minimum spanning tree induced by the clusters plus the edges connecting two D-odd vertices in the same cluster.

It is possible to formulate the GARP using an integer program with two sets of binary variables. For each $e \in E$, let x_e and y_e be binary variables associated with the first and second traversal of edge e. Specifically, $x_e = 1$ means that edge e is crossed in the solution tour, while $y_e = 1$ would imply that solution tour crossed twice the edge e. The formulation is as follows:

$$(F_{xy}) \quad \min \sum_{e \in E} c_e(x_e + y_e) \tag{1}$$

$$x(D_k) \geq 1 \qquad k \in K \tag{2}$$

$$(x+y)(\delta(S)) \geq 2 \qquad S = \cup_{k \in K_S} V_k, \quad K_S \subseteq K \tag{3}$$

$$(x-y)(\delta(S) \setminus F) + y(F) \geq x(F) - |F| + 1 \qquad S \subset V,\ F \subseteq \delta(S), |F| \text{ odd} \tag{4}$$

$$y_e \leq x_e, \qquad e \in E \tag{5}$$

$$x_e, y_e \in \{0,1\},\ e \in E \tag{6}$$

Inequalities (2) guarantee that at least one edge of each cluster is traversed, whereas connectivity with the depot is implied by constraints (3). Constraints (4) are an adaptation to the GARP of co-circuit inequalities [9], which ensure even degree of the visited vertices with respect to the solution. Broadly speaking, they impose that if a solution uses an odd number of edges incident with a set of vertices S, the solution uses at least one additional edge of the cut-set of S. They further exploit the precedence relationship of the x variables with respect to the y variables, which is captured by inequalities (5). Constraints (4), which were proposed by [15] for the Maximum Benefit Chinese Postman Problem, are a reinforcement of those used in [4, 7] for the Clustered Prize-collecting Arc Routing Problem and the Privatized Rural Postman problem, respectively. Formulation (1)–(6) involves $2|E|$ variables and a number of constraints of types (3) and (4) which is exponential on $|V|$. Both families of inequalities can be separated in polynomial time [10, 4, 7, 30].

We next prove that in formulation (1)–(6) the integrality condition on the y variables can be relaxed, and still obtain optimal solutions with binary values for the y variables.

Proposition 2.2 *Let (x^*, y^*) be an optimal solution to formulation (1)–(5) where constraints (6) have been relaxed to $x^*_e \in \{0,1\}$, $0 \leq y^*_e \leq 1$, $\forall e \in E$. Then, $y^*_e \in \{0,1\}$, $\forall e \in E$.*

Proof: Define $G_y = (V_y, E_y)$ where $E_y = \{e \in E : 0 < y^*_e < 1\}$, and suppose that it is not empty. Then, it must contain no tours, since otherwise, let $T \subseteq E_y$ denote such a tour, and $\Delta = \min\{y^*_e : e \in T\}$. The solution $x^s = x^*$, and $y^s_e = y^*_e - \Delta$, $\forall e \in T$ plus $y^s_e = y^*_e$ otherwise, is feasible for (2)–(5) with the integrality conditions on the x variables, and its value is at least as good as that of (x^*, y^*).

Therefore, if E_y is non-empty there exists $u \in V_y$ which is a leaf in G_y. Let $f = (u, v)$ be the only edge of E_y incident with u, and consider the sets $S = \{u\}$ and $F = \{e \in \delta(u) : x^*_e > y^*_e\}$.

By definition, $f \in F \subseteq \delta(S)$. By constraints (5), $x^*_e = 1$ for all $e \in F$, so $x^*(F) = |F|$. Since f is the only edge of E_y incident with u, we have $y^*_e = 0$ for all $e \in F \setminus \{f\}$ and $y^*(F) = y^*_f$. Furthermore, $(x^* - y^*)(\delta(S) \setminus F) = 0$, because $x^*_e = y^*_e$ for all $e \in \delta(S) \setminus F$.

Consider the following two cases:

(a) $|F|$ odd. The left hand side of the constraint (4) associated with S and F takes the value $(x^* - y^*)(\delta(S) \setminus F) + y^*(F) = y^*_f$, whereas the right hand side of the constraint takes the value $x^*(F) - |F| + 1 = |F| - |F| + 1$, so the constraint is violated.

(b) $|F|$ even. Now $|F| \geq 2$, since $f \in F$. Thus, $\emptyset \neq F' = F \setminus \{f\} \subset \delta(S)$ and $|F'|$ is odd. Hence, the constraint (4) associated with S and F' must hold. Its right hand side takes the value $x^*(F') - |F'| + 1 = 1$, whereas in the left hand we have $(x^* - y^*)(\delta(S) \setminus F') = 1 - y^*_f < 1$, and $y^*(F') = 0$. Thus the constraint is violated.

As a consequence, if (x^*, y^*) is optimal, $E_y = \emptyset$. Hence, $y^*_e \in \{0, 1\}$, $\forall e \in E$. ■

3 Dominance conditions and improved formulation for the GARP

Recall that E contains the edges in D plus additional non-demand edges connecting every pair u, v of vertices not connected with an edge of D, with cost c_{uv} equal to that of the shortest path in the original graph. Therefore:

Lemma 3.1 *For any non-demand edge $(u, v) \in R$, $c_{uv} \leq c_{us} + c_{sv}$ for all $(u, s), (v, s) \in E$*

Proof: Since all costs for non-demand edges (u, v) are defined as equal to that of the shortest path between u and v in the original graph, triangle inequality holds for the costs of such edges by definition. ■

Lemma 3.2 *For any demand edge, $(u, v) \in D$, traversed in an optimal GARP solution it holds that $c_{uv} \leq c_{us} + c_{sv}$ for all $(u, s), (v, s) \in D$*

Proof: No optimal GARP solution exists that traverses a demand edge $(u, v) \in D$ with $c_{uv} > c_{us} + c_{sv}$, for some $(u, s), (v, s) \in D$. Otherwise substituting (u, v) by the pair of edges $(u, s), (v, s)$ would give a feasible solution with a better value. ■

Theorem 3.1 *An optimal GARP solution exists satisfying the following properties:*

(a) Exactly one demand edge of each cluster is traversed.

(b) No consecutive non-demand edges are traversed.

(c) No edge is traversed twice.

Proof: This proof would consist of repeatedly applying the triangle inequality. ■

Note that without the vertex-disjoint assumption on the clusters, the above first assertion would not necessarily hold.

Therefore, we can eliminate from G all demand edges whose costs do not satisfy the triangle inequality of Lemma 3.2. By property (b) we can further simplify G by removing any non-demand edge within a cluster. Thus, non-demand edges either connect vertices in different clusters or are incident with the depot. That is, E contains the edges in D whose costs satisfy the property of Lemma 3.2, plus additional non-demand edges representing shortest paths in the original graph, which connect the depot with any other vertex and every pair of vertices in different clusters.

In the simplified graph we denote $n = |V|$, $m = |E|$, and $p = |K|$. Let also $R = E \setminus D$, and $R_d = \delta(v_d)$.

Assuming that *simple* tours are those in which no edge is crossed twice, we call *simple alternating tour* to a simple tour through the depot with an odd number of edges alternating between demand and non-demand, except for the two edges incident with the depot, which are both non-demand. We call *alternating tour* to the union of (possibly only one) simple alternating tours which are vertex disjoint for all vertices but the depot. Any alternating tour which traverses a demand edge of each cluster is a feasible solution to the GARP. A simple alternating tour traversing a demand edge of each cluster is called *alternating circuit*. Note that alternating circuits exist since any set of p demand edges from different clusters, can be completed into such a solution as non-demand edges exist connecting any pair of vertices in different clusters.

Over the property (c) we can build an improved formulation for the GARP which only uses one set of binary variables to indicate the edges that are traversed in the alternating tours. This is, let us define variable x_e meaning whether the solution tour uses the edge e. Then, the formulation is as follows:

$$(F_x) \quad \min \sum_{e \in E} c_e x_e \tag{7}$$

$$x(\delta_D(v)) = x(\delta_R(v)) \qquad v \in V \setminus \{v_d\} \tag{8}$$

$$x(D_k) = 1 \qquad k \in K \tag{9}$$

$$x(\delta(S)) \geq 2 \qquad S = \cup_{k \in K_S} V_k, \quad K_S \subseteq K, |K_S| \geq 2 \tag{10}$$

$$x_e \in \{0,1\}, \ e \in E \tag{11}$$

Constraints (8) play a double role. On the one hand, they ensure that feasible tours alternate between demand and non-demand edges. On the other hand, they guarantee that in any solution all vertices have even degree: two in the case of visited vertices and zero otherwise. The condition that exactly one demand edge of each cluster is traversed, is enforced by equalities (9). Connectivity with the depot of solutions is implied by constraints (10). These constraints are imposed for sets S consisting of at least two clusters, since for each $k \in K$ the equality (9) together with constraints (8) already imply that $x(\delta(V_k)) = 2$. Note that no constraint on the degree of the depot is imposed. This is, feasible solutions to (7)–(11) are not necessarily alternating circuits as they may also be alternating tours.

Formulation (7)–(11) involves m variables, $n-1$ constraints (8) and p constraints (9). The number of constraints of types (10) is exponential on n.

Remark 3.1 Inequalities (10) can be written as $\delta_R(S) \geq 2$, because $\delta_D(S) = \emptyset$ for all $S = \cup_{k \in K_S} V_k, K_S \subseteq K$.

It is clear that if x is feasible for formulation F_x then (x, y) with $y_e = 0$ for all $e \in E$ is also feasible for F_{xy}. That is, the feasible domain for F_x is contained in the projection of the feasible domain for F_{xy} onto the subespace defined by equations $\{y_e = 0, e \in E\}$. The same applies to their linear programming (LP) relaxations. Therefore, we have:

Proposition 3.3 *Let* $XY = \{(x,y) \in \{0,1\}^m \times \{0,1\}^m \mid (x,y)$ *satisfy* $(2)-(5)\}$ *and* $X = \{x \in \{0,1\}^m \mid x$ *satisfies* $(8)-(10)\}$ *denote the domains of formulations of* F_{xy} *and* F_x. *Let also* $\overline{XY}$ *and* $\overline{X}$ *denote the domains of their respective LP relaxations, and* $\overline{z}_{xy}$ *and* $\overline{z}_x$ *the optimal LP values for* F_{xy} *and* F_x. *Then,*

- $X \subseteq XY \cap \{y_e = 0, e \in E\}$.
- $\overline{X} \subseteq \overline{XY} \cap \{y_e = 0, e \in E\}$.
- $\overline{z}_{xy} = \overline{z}_x$.

4 The polyhedron associated with F_x

Let P be the convex hull of feasible solutions to formulation F_x:

$$P = conv\{x \in \{0,1\}^m \mid x \text{ satisfies } (8) - (10)\}.$$

Theorem 4.1 *$dim(P) = m - (n-1) - p$, where $m = |E|, n = |V|$ and $p = |K|$.*

Proof: The $(n-1)+p$ equality constraints (8) and (9) are linearly independent. Thus, $dim(P) \leq m-(n-1)-p$. To see that $dim(P) = m-(n-1)-p$ we will find $m-(n-1)-p+1$ affinely independent points of P.

Let x^C be the incidence vector of an alternating circuit C. For ease of notation we suppose that in C the clusters are visited in the natural order $1, \ldots, p$. Thus, C can be expressed as $v_d - u_1 - v_1 - \cdots - u_k - v_k - \cdots - u_p - v_p - v_d$ with $(u_k, v_k) \in D_k$, $k \in K$ and $(v_{k-1}, u_k) \in R$. We will find a point associated with each edge in $E \setminus R_d$, except for the demand edges of C.

Consider the following cases:

case (a): $e \in D^0 = \{e \in D \mid x^C_e = 0\}$.

Let us denote such an edge $e = (\bar{u}_k, \bar{v}_k) \in D_k$, for some $k \in K$.

Define x^* as the incidence vector of the alternating tour which differs from C in that the chain $v_{k-1} - u_k - v_k - u_{k+1}$ is substituted by the chain $v_{k-1} - v_d - u_{k+1}$. Define also x' as the incidence vector of the simple alternating tour $v_d - \bar{u}_k - \bar{v}_k - v_d$. Then, $x^* + x' \in P$, as shown in Figure 2. In Figures 2, 3 and 4, edges traversed by C are in black, non-traversed edges in light gray, dashed lines mean paths with possibly more than one edge, and clusters are represented by thick lines.

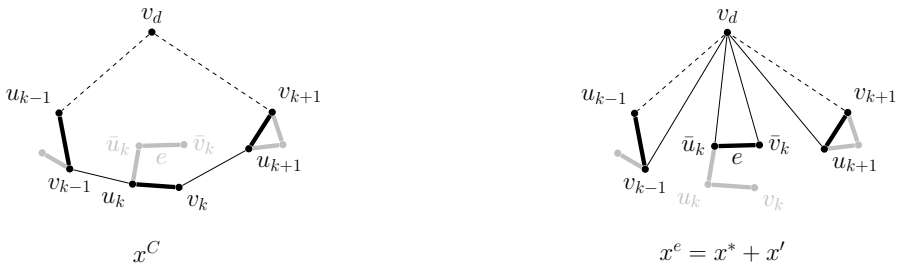

Figure 2: *Case (a) in proof of Theorem 4.1.*

In order to construct the final matrix, we will denote by $x^e = x^* + x'$ the solutions built this way.

Note that there are $d_0 = |D^0| = |D| - p$ such points.

case (b): $e \in R^0 = \{e \in R \setminus R^d \mid x^C_e = 0\}$.

Then, $e = (\bar{v}_r, \bar{u}_s)$ with $\bar{v}_r \in V_r$ and $\bar{u}_s \in V_s$ for some $r, s \in K$, $r < s$. Define x^* as the incidence vector of the alternating tour which differs from C in:

(*i*) The chain $v_{r-1} - u_r - v_r - u_{r+1}$ is replaced by the chain $v_{r-1} - v_d - u_{r+1}$;

(*ii*) The chain $v_{s-1} - u_s - v_s - u_{s+1}$ is replaced by the chain $v_{s-1} - v_d - u_{s+1}$.

In addition, select $\bar{u}_r \in \delta_D(\bar{v}_r)$, and $\bar{v}_s \in \delta_D(\bar{u}_s)$. Define x' as the incidence vector of the simple alternating tour $v_d - \bar{u}_r - \bar{v}_r - \bar{u}_s - \bar{v}_s - v_d$.

We will refer to these solutions by $y^e = x^* + x' \in P$. A generic point of this type might be seen in Figure 3.

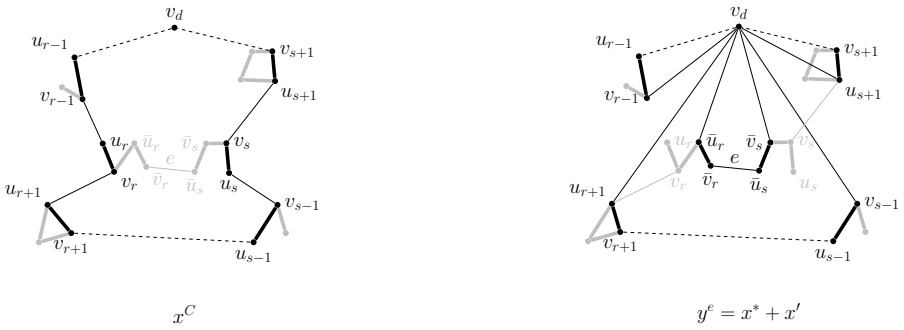

Figure 3: *Case (b) in proof of Theorem 4.1.*

Note that possibly $\bar{v}_r$ is u_r or v_r, and also that $\bar{u}_s$ might coincide with u_s or v_s.

The quantity of points in this case is $r_0 = |R^0|$.

case (c): $e \in R^1 = \{e \in R \setminus R^d \mid x^C_e = 1\}$.

In the case of $e = (v_{k-1}, u_k)$ for some $k \in K \setminus \{1\}$, define x^* as the incidence vector of the alternating tour that results from C when edge (v_{k-1}, u_k) is substituted by the chain $v_{k-1} - v_d - u_k$.

Let us denote by $z^e = x^* \in P$ the solutions built this way. In Figure 4 an example of one of this points is illustrated.

Figure 4: *Case (c) in proof of Theorem 4.1.*

The number of points provided by case (c) is $r_1 = |R^1| = p - 1$.

There are in total $|R \setminus R^d| = (m - |D|) - (n - 1)$ points of types (b) plus (c). And thus, the total number of points is $m - (n - 1) - p$.

All points considered so far are affinely independent. To build the corresponding matrix, we firstly partition the set of edges in E by the sets $E = D^0 \cup D^1 \cup R^0 \cup R^1 \cup R^d$ with $D^1 = \{e \in D \mid x^C_e = 1\}$ and $d_1 = |D^1|$. The first two column blocks correspond to demand edges in D^1 and D^0. The next two, to edges in R^0 and R^1. And the last, to those of R^d. The first row is associated with x^C whereas the remaining rows are associated with points x^e, y^e and z^e. In the following matrices, the blocks denoted by A have 0-1 entries but no particular structure. Their dimensions are indicated in the subindex while the superindex denotes the kind of points that compose them. $\mathbb{0}$ and $\mathbb{1}$ denote matrices of appropriate dimensions with all-zero and all-one entries respectively, and $\mathbb{I}$ the identity matrix.

	D^1	D^0	R^0	R^1	R^d
x^C	$\mathbb{1}_{1\times p}$	$\mathbb{O}_{1\times d_0}$	$\mathbb{O}_{1\times r_0}$	$\mathbb{1}_{1\times r_1}$	$1 \quad 0 \quad \dots \quad 0 \quad 1$
x^e	$A^x_{d_0\times p}$	$\mathbb{I}_{d_0\times d_0}$	$\mathbb{O}_{d_0\times r_0}$	$A^x_{d_0\times r_1}$	$A^x_{d_0\times(n-1)}$
y^e	$A^y_{r_0\times p}$	$A^y_{r_0\times d_0}$	$\mathbb{I}_{r_0\times r_0}$	$A^y_{r_0\times r_1}$	$A^y_{r_0\times(n-1)}$
z^e	$\mathbb{1}_{(p-1)\times p}$	$\mathbb{O}_{(p-1)\times d_0}$	$\mathbb{O}_{(p-1)\times r_0}$	$(\mathbb{1}\text{- }\mathbb{I})_{(p-1)\times r_1}$	$A^z_{(p-1)\times(n-1)}$

Matrix 1

By subtracting the first row to each of the rows in the last block associated with z^e we obtain

	D^1	D^0	R^0	R^1	R^d
x^C	$\mathbb{1}_{1\times p}$	$\mathbb{O}_{1\times d_0}$	$\mathbb{O}_{1\times r_0}$	$\mathbb{1}_{1\times r_1}$	$1 \quad 0 \quad \dots \quad 0 \quad 1$
x^e	$A^x_{d_0\times p}$	$\mathbb{I}_{d_0\times d_0}$	$\mathbb{O}_{d_0\times r_0}$	$A^x_{d_0\times r_1}$	$A^x_{d_0\times(n-1)}$
y^e	$A^y_{r_0\times p}$	$A^y_{r_0\times d_0}$	$\mathbb{I}_{r_0\times r_0}$	$A^y_{r_0\times r_1}$	$A^y_{r_0\times(n-1)}$
z^e	$\mathbb{O}_{(p-1)\times p}$	$\mathbb{O}_{(p-1)\times d_0}$	$\mathbb{O}_{(p-1)\times r_0}$	$-\mathbb{I}_{(p-1)\times r_1}$	$\widehat{A}^z_{(p-1)\times(n-1)}$

Matrix 2

Now we can use the rows in the block associated with z^e to cancel out the elements in $A^x_{d_0\times r_1}$ and $A^y_{r_0\times r_1}$ and obtain Matrix 3 which is clearly is of full rank.

	D^1	D^0	R^0	R^1	R^d
x^C	$\mathbb{1}_{1\times p}$	$\mathbb{O}_{1\times d_0}$	$\mathbb{O}_{1\times r_0}$	$\mathbb{1}_{1\times r_1}$	$1 \quad 0 \quad \dots \quad 0 \quad 1$
x^e	$A^x_{d_0\times p}$	$\mathbb{I}_{d_0\times d_0}$	$\mathbb{O}_{d_0\times r_0}$	$\mathbb{O}_{d_0\times r_1}$	$\widehat{A}^x_{d_0\times(n-1)}$
y^e	$A^y_{r_0\times p}$	$A^y_{r_0\times d_0}$	$\mathbb{I}_{r_0\times r_0}$	$\mathbb{O}_{r_0\times r_1}$	$\widehat{A}^y_{r_0\times(n-1)}$
z^e	$\mathbb{O}_{(p-1)\times p}$	$\mathbb{O}_{(p-1)\times d_0}$	$\mathbb{O}_{(p-1)\times r_0}$	$-\mathbb{I}_{(p-1)\times r_1}$	$\widehat{A}^z_{(p-1)\times(n-1)}$

Matrix 3

■

Theorem 4.2

- *For all $e \in R$ $x_e \geq 0$, is a facet of P.*
- *For all $e \in D_k$, $k \in K$ with $|D_k| \geq 2$, $x_e \geq 0$, is a facet of P.*
- *For all $e = (v_d, u) \in R^d$, with $u \in V_k$, $k \in K$, $|D_k| \geq 2$, $x_e \geq 0$, is a facet of P.*

Proof: A slight modification of the above proof can be used to prove that non-negativity inequalities are facets of P.

To see that $x_e \geq 0$ is a facet for $e \in R \setminus R^d$, the initial solution x^C must be chosen in such a way that $x^C_e = 0$, i.e. e is one of the edges of case (b). Then, the additional points are defined as above with the only exception of the point associated with edge e, which is not used, thus obtaining in total $m-(n-1)-p$ affinely independent points satisfying $x_e = 0$. When $e \in D_k$ for some $k \in K$, we need the additional assumption that D_k contains at least one more edge, since otherwise all feasible solutions satisfy $x_e = 1$. Then, to prove that when $|D_k| \geq 2$, $x_e \geq 0$ with $e \in D_k$ is a facet of P we again chose an initial solution with $x^C_e = 0$, so e is one of the edges of case (a). Now, the additional $m-n-p$ affinely independent points are also defined as in the proof of Theorem 4.1 with the only exception of the point associated with edge e, which is not used. When $e \in R^d$, with $e = (v_d, u_1) \in R^d \cap \delta_R(V_1)$ and $|D_1| \geq 2$, the above initial solution and affinely independent points for any edge $e' = (u_1, v_1) \in D_1$ can also be used. Note that any initial solution with $x^C_{e'} = 0$ also satisfies that $x^C_e = 0$, and any set of $m-(n-1)-p$ affinely independent points satisfying $x_{e'} = 0$, also satisfy $x_e = 0$. When $e = (v_d, u) \in \delta_R(V_k)$ for some $k > 1$ with $|D_k| \geq 2$, we proceed similarly with the only difference that the initial solution x^C must be chosen in such a way that cluster k is visited in the first place. Therefore, we have the following result:

Theorem 4.3 *Connectivity inequalities $x(\delta(S)) \geq 2$, with $S = \cup_{k \in K_S} V_k, K_S \subseteq K$, $|K_S| \geq 2$ are facets of P.*

Proof: The proof of this theorem would be similar to that of Theorem 4.2 but we avoid it here for the sake of brevity.

5 Valid inequalities

In this section we present several families of valid inequalities that can be used to reinforce the LP relaxation of formulation (7)–(11). These families can be classified in three different types: connectivity, parity and matching. For each type of inequalities we first present an adaptation to the GARP of some well-known family, and then we introduce a new family, which reinforces and extends the original one. In particular we introduce stronger connectivity inequalities, parity inequalities which generalize co-circuit inequalities [9], as well as matching-type inequalities which generalize classical inequalities for odd subsets of vertices [21]. The section closes by relating the generalized co-circuit inequalities to the generalized matching-type inequalities.

5.1 Connectivity Inequalities

In formulation F_x connectivity of each cluster with the depot is implied by equalities (9) together with constraints (8), whereas connectivity with the depot of vertex subsets consisting of at least two clusters is implied by constraints (10). When integrality conditions are relaxed, additional connectivity inequalities can be used to reinforce the LP relaxation of F_x.

Observe first that constraints (10) are valid for a larger family of sets S than that defined by the union of vertex sets of several clusters. In particular, we know that in any feasible solution any set containing all the vertices of some cluster will be visited. Therefore, its cut-set must be at least 2. That is, when $V \setminus \{v_d\} \supseteq S \supseteq V_k$ for some $k \in K$ the associated constraint (10) is valid for F_x even if S is not necessarily the union of the vertex sets of several clusters. We call *cluster-connectivity* constraints to this extended family of inequalities (10).

Other families of connectivity constraints are aso valid for any vertex set, even if it does not fully contain the set of vertices of any cluster. In particular,

$$x(\delta(S)) \geq 2x_e \qquad S \subseteq V \setminus \{v_d\},\ e \in E(S) \tag{12}$$

are valid for F_x since they impose that the cut-set of any visited set of vertices is traversed at least twice. Similar inequalities are well-known for other arc routing problems in which the set of vertices to be visited by a certain route is not known in advance (see, for instance, [10, 4]). We call *set-connectivity* to the family of inequalities (12). For sets S that do not fully contain any cluster vertex set, these inequalities are not implied by the cluster-connectivity constraints. Thus, we will consider set-connectivity constraints associated with sets $S \subseteq V \setminus \{v_d\}$ such that $V_k \setminus S \neq \emptyset$ for all $k \in K$.

Observe that the set-connectivity constraint associated with $e \in E(S)$ for any set $S \subseteq V \setminus \{v_d\}$ can be reinforced because of the alternating condition of optimal tours:

- When $e \in D_k$ for some k and $x_e = 1$, no other edge in D_k can be at value one. This has a double effect. First, we can reinforce its right hand side and obtain the strengthened constraint

$$x(\delta(S)) \geq 2 \sum_{e \in E(S) \cap D_k} x_e. \tag{13}$$

  Now, for a given S and k with $V_k \setminus S \neq \emptyset$, all the edges in $E(S) \cap D_k$ yield the same constraint (13). Constraints (13) are called *D-set-connectivity* inequalities.

  We can further reinforce constraint (13) by eliminating some terms in the left hand side. When some edge in $E(S) \cap D_k$ is at value one, then the only edges in $\delta(V_k \cap S)$ that can take value one are non-demand ones. Thus, inequalities (13) can be reinforced to:

$$x(\delta_D(S \setminus V_k)) + x(\delta_R(S)) \geq 2 \sum_{e \in E(S) \cap D_k} x_e. \tag{14}$$

  Constraints (14), the reinforced family of inequalities (13), are called *D^+-set-connectivity* constraints.

- When $e \in R$, i.e. $e \in \delta(V_k : V_l)$ for some $k, l \in K$, and $x_e = 1$, we can proceed similarly. First, no other non-demand edge in $(V_k : V_l) \cap E(S)$ can be used, so the right hand side of (12) can be reinforced to

$$x(\delta(S)) \geq 2 \sum_{e \in E_R(S) \cap (V_k : V_l)} x_e. \tag{15}$$

  Constraints (15) are called *R-set-connectivity* inequalities. Now, when some non-demand edge in $(V_k : V_l) \cap E_R(S)$ is used, no other non-demand edge in $(V_k : V_l)$ can be used. That is, for $S \subseteq V \setminus \{v_d\}$ and $k, l \in K$, with $V_k \setminus S \neq \emptyset$, $V_l \setminus S \neq \emptyset$ and $E_R(S) \cap (V_l : V_l) \neq \emptyset$ the inequality (15) can be reinforced to:

$$x(\delta_D(S) + x(\delta_R(S) \setminus (V_k : V_l)) \geq 2 \sum_{e \in E_R(S) \cap (V_k : V_l)} x_e. \tag{16}$$

  Constraints (16), the reinforced family of inequalities (15), are called *R^+-set-connectivity* constraints.

5.2 Parity inequalities: Generalized Co-circuit Inequalities

For binary solutions, constraints (8) guarantee the parity of each vertex as well as of each vertex set. However, when integrality is relaxed, even if the parity of each vertex is still guaranteed, the parity of subsets of vertices may no longer hold. Therefore, co-circuit inequalities,

$$x(\delta(S) \setminus F) \geq 1 - |F| + x(F) \qquad S \subset V,\ F \subseteq \delta(S), |F| \text{ odd} \tag{17}$$

can be used to cut-off such solutions. Inequalities (17) can be extended to stronger valid inequalities for the GARP, as we next see.

Theorem 5.1 *Let $S \subset V$ be a given set of vertices and $F \subseteq \delta(S)$, $F = F_1 \cup \cdots \cup F_r$, with $F_i \cap F_j = \emptyset$, $i \neq j$, and r odd and such that every feasible GARP solution satisfies $x(F_i) \leq 1$, $i = 1, \ldots, r$. Then, the following Generalized Co-circuit Inequality (GCI) is valid for F_x:*

$$x(\delta(S) \setminus F) \geq 1 - r + x(F). \tag{18}$$

Proof: Let x denote the incidence vector of a feasible solution to F_x. If $1-r+x(F) \leq 0$ the inequality trivially holds. Otherwise, $1 - r + x(F) > 0$ or, taking into account the integrality of the solution, $1 - r + x(F) \geq 1$, so $x(F) \geq r$. On the other hand, by hypothesis, $x(F_i) \leq 1$, $i = 1, \ldots, r$, so we have $x(F) = \sum_{i=1}^{r} x(F_i) \leq r$, where the first equality follows since $F_i \cap F_j = \emptyset$, $i \neq j$. Thus, $x(F) = r$ (i.e. $x(F_i) = 1$, $i = 1, \ldots, r$) and the right hand side of the inequality takes the value 1. Since r is odd, the tour must traverse at least one additional edge in the cut-set of S. Given that the solution already traverses one edge of each F_i and the edges in each F_i are mutually incompatible, the additional traversed edge must belong to $\delta(S) \setminus F$, and the inequality holds. ■

Remark 5.1

- Observe that the hypotheses of Theorem 5.1 are natural in the context of the GARP. For instance, for a given $S \subset V$ if we define $F_i = \delta_D(S \cap V_i)$, $i \in K$, it holds that $F_i \cap F_j = \emptyset$, for all $i \neq j$ and $x(F_i) \leq 1$ in every feasible GARP solution.
- It is clear that when $|F_i| = 1$, $i = 1, \ldots, r$, GCIs reduce to co-circuit inequalities (17). However, for the general case when $|F_i| > 1$ for some i, the GCI dominates the classical co-circuit inequality. Note that, in fact, when $|F_i| > 1$ for some i the classical co-circuit inequality will never be violated, since $1 - |F| + x(F) \leq 1 - |F| + r \leq 0$, because $r < |F|$.
- The GCI (18) is valid even if the F_i's are not pairwise disjoint. In this case, if $x_e = 1$ for some $e \in F_i \cap F_j$, $i \neq j$, then $x(F) < r$, so the GCI is not tight.

5.3 Matching type inequalities: Generalized Matching Inequalities

Another family of valid inequalities can be derived from the fact that, in alternating circuits, non-demand edges non-incident with the depot define matchings on the graph induced by the visited vertices excluding the depot.

Theorem 5.2 *Let $S = \{v_1, \ldots v_r\} \subset V \setminus \{v_d\}$, and r odd. Then the following inequality is valid for P:*

$$x(E_R(S)) \leq \frac{|S| - 1}{2}. \tag{19}$$

Proof: Since any feasible solution alternates between demand and non-demand edges, the set of non-demand edges is a matching on the subgraph induced by the set of visited vertices but the depot. Therefore, (19) is valid. ■

In the following, inequalities (19) will be referred to as *Matching Inequalities*

The set of demand edges traversed in a feasible tour also defines a perfect matching in the graph induced by visited vertices but the depot. Thus, even though similar inequalities applied to $S \subseteq V_k$ would be also valid for the GARP, they have not been introduced because inequalities 9 are stronger.

When several vertices of S belong to the same cluster, inequality (19) may be very loose. Thus one may wonder if valid inequalities of this type can be obtained, in which the right hand side depends on the number

of clusters involved in set S, say r. Note that when several vertices of S belong to the same cluster, the tighter inequality $x(E_R(S)) \leq \lfloor \frac{r}{2} \rfloor$ need not be valid, as illustrated in Figure 5 for $S = \{u_1, u_2, u_3, u_4\}$.

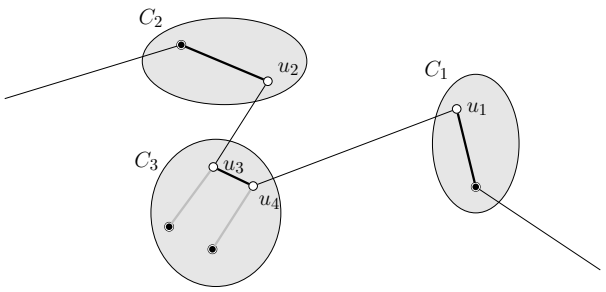

Figure 5: *Rationale for Generalized Matching Inequalities.*

In particular, two non-demand edges may connect clusters C_1, C_2 an C_3. This is posible, because the depicted solution traverses demand edge (u_3, u_4) connecting two vertices of S.

Therefore, if S contains a subset of vertices S_i, all of them belonging to the same cluster and no demand edge connecting two vertices of S_i is used, then the cut-set $x(\delta(S_i))$ may contain at most one non-demand edge. However, if a demand edge connecting two vertices of S_i is used, then the cut-set $x(\delta(S_i))$ may contain up to two non-demand edges. This idea is formalized below.

Theorem 5.3 *Let $S \subset V \setminus \{v_d\}$ be a set such that $S = S_1 \cup \cdots \cup S_r$ with $S_i \subseteq V_i$, and $r < p$ odd. Then, the following Generalized Matching Inequality (GMI) is valid for the GARP:*

$$x(E_R(S)) \leq \frac{r-1}{2} + x(E_D(S)). \tag{20}$$

Proof: Let x denote the incidence vector of a feasible solution to F_x.

When $x(E_D(S)) = 0$ inequality (20) is valid since it must hold that $x(\delta_R(S_i)) \leq 1$, $i = 1, \ldots, r$.

Let us then assume that $x(E_D(S)) > 0$, and rewrite inequality (20) as

$$2x(E_R(S)) \leq r - 1 + 2x(E_D(S)).$$

We have

$$\sum_{u \in S} x(\delta_R(u)) = 2x(E_R(S)) + x(\delta_R(S)),$$

and

$$\sum_{u \in S} x(\delta_D(u)) = 2x(E_D(S)) + x(\delta_D(S)).$$

Therefore, by constraints (8), $2x(E_R(S)) + x(\delta_R(S)) = 2x(E_D(S)) + x(\delta_D(S))$, so

$$\begin{aligned} 2x(E_R(S)) = &\ 2x(E_D(S)) + x(\delta_D(S)) - x(\delta_R(S)) \leq \\ \leq &\ 2x(E_D(S)) + x(\delta_D(S)). \end{aligned} \tag{21}$$

On the other hand, since $S_i \subseteq V_i$, we have

- $x(E_D(S_i)) + x(\delta_D(S_i)) \leq x(D_i) = 1$, and
- $x(\delta_D(S_i : S_j)) = 0$, $i \neq j$.

Therefore, $x(E_D(S)) + x(\delta_D(S)) \leq \sum_{i=1}^{r} [x(E_D(S_i)) + x(\delta_D(S_i))] \leq r$.
As a consequence, $x(\delta_D(S)) \leq r - 1$ because we are assuming that $x(E_D(S)) > 0$.

Finally, from (21) we have

$$2x(E_R(S)) \leq 2x(E_D(S)) + x(\delta_D(S)) \leq 2x(E_D(S)) + r - 1,$$

and the result follows. ■

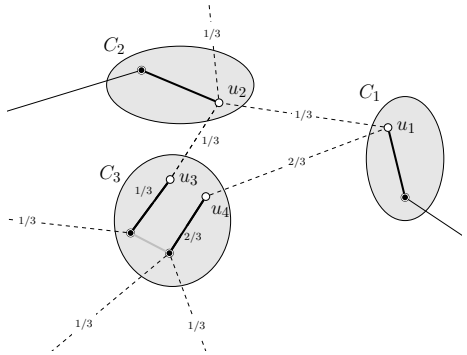

Figure 6: *Violated Generalized Matching Inequality.* $S_1 = \{u_1\}$, $S_2 = \{u_2\}$, $S_3 = \{u_3, u_4\}$

Figure 6 gives an example of a GMI violated by a (fractional) solution to the linear programming relaxation of F_x.

As a particular case of GMIs, when $|S| = r$ we obtain the matching inequalities (19), since in this case $|S_i| = 1$, $i = 1, \ldots, r$ so $E_D(S) = \emptyset$ and thus $x(E_D(S)) = 0$.

The following result relates GCIs and GMIs. We see that they are equivalent with respect to the domain of feasible solutions to the LP relaxation of F_x.

Theorem 5.4 *Let $S = S_1 \cup \cdots \cup S_r \subset V \setminus \{v_d\}$ be a given set of vertices with $S_i \subseteq V_i$, $i = 1, \ldots, r$, and $r < p$ odd. Define the set $F = F_1 \cup \cdots \cup F_r$ with $F_i = \delta_D(S_i)$. Then, any feasible solution to the relaxed problem, x, violates the generalized matching inequality if and only if the generalized co-circuit inequality associated with F is violated by x.*

Proof: By definition of F_i, $x(F_i) \leq x(D_i) \leq 1$.

In addition,

$$x(F) = \sum_{i=1}^{r} x(F_i) = \sum_{i=1}^{r} x(\delta_D(S_i)) = x(\delta_D(S)) \qquad \text{thus} \quad x(\delta(S) \setminus F) = x(\delta_R(S)).$$

By constraints (8), $\sum_{u \in S} x(\delta_R(u)) = \sum_{u \in S} x(\delta_D(u))$.

We also have

$$\sum_{u \in S} x(\delta_R(u)) = 2x(E_R(S)) + x(\delta_R(S)),$$

and

$$\sum_{u \in S} x(\delta_D(u)) = 2x(E_D(S)) + x(\delta_D(S)).$$

Therefore, $2x(E_R(S)) + x(\delta_R(S)) = 2x(E_D(S)) + x(\delta_D(S)),$ so

$$x(\delta_R(S)) = 2x(E_D(S)) + x(\delta_D(S)) - 2x(E_R(S)).$$

Thus,

$$x(\delta(S) \setminus F) = x(\delta_R(S)) = 2x(E_D(S)) + x(\delta_D(S)) - 2x(E_R(S)). \tag{22}$$

On the other hand, $1 - r + x(F) = 1 - r + x(\delta_D(S))$, which by adding and subtracting $2x(E_D(S))$ can be rewritten as

$$1 - r + x(F) = 2x(E_D(S)) + x(\delta_D(S)) - 2\left(\frac{r-1}{2} + x(E_D(S))\right). \tag{23}$$

By (22) and (23) we have that the GMI associated with S is violated by x, if and only if $x(E_R(S)) > \frac{r-1}{2} + x(E_D(S))$, and the result follows. ■

In the simpler case when all vertices in S belong to different clusters, $E_D(S) = \emptyset$. So, if we define $F_i = \delta_D(v_i)$, $i = 1, \ldots, r$, we have the following:

Corollary 5.1 *The matching inequality associated with S is violated by a feasible solution x if and only if the generalized co-circuit inequality associated with F is also violated by x.*

6 Separation of inequalities

Next we describe separation algorithms for the different families of inequalities presented above. Throughout this section $x \in [0,1]^m$ denotes the vector we want to separate. And from now on, let $G = (V, E)$ denote the support graph of x. This means that in the following, E are the subset of edges from the original graph with $x_e > 0$, and $V = V(E)$ the subset of vertices incident with some of those edges.

6.1 Connectivity inequalities

The following separation procedure is exact and similar to the one used by other authors to separate constraints (12) for other arc routing problems [12, 3, 4, 13]. It all starts by computing a graph $G_k(V_k, E_k)$ with the depot, and a vertex for each cluster in the original graph. Weighted edges E_k are defined with the sum of all x corresponding to edges connecting each couple of clusters in the original graph, and doing so with the depot.

Each connected component of G_k but the one with the depot defines a violated inequality (10).

When G_k is connected, the process follows by computing a mincut tree to check whether the value of the mincut is smaller than two minus an $\epsilon = 0.05$. If found, the set of vertices not containing the depot reveals a new violated inequality (10).

Once no violated inequality is found with the heuristic procedure tried so far, the process starts again with G instead of G_k. This leads to find out if some inequality of type (13) or (15) is violated by the current x.

6.1.1 D-set-connectivity and R-set-connectivity constraints

The same procedure above can be used to separate D-set-connectivity and R-set-connectivity constraints (13) and (15) and their reinforced versions (14) and (16). For each edge $e \in E$, in the solution graph, we identify the minimum cut between the depot and edge e, $\delta(S)$, and we consider two different cases.

Firstly, for each cluster with some edge in $E(S)$ we check if the associated D-set-connectivity inequality (13) is violated. That is, whether

$$x(\delta(S)) < 2 \sum_{e \in D_k \cap E(S)} x_e.$$

If so, the reinforced D^+-set-connectivity inequality (14) is also violated. Otherwise, an additional check is done to see whether it is the reinforced D^+-set-connectivity inequality (14) which is violated,

$$x(\delta_D(S \setminus V_k)) + x(\delta_R(S)) < 2 \sum_{e \in D_k \cap E(S)} x_e.$$

After that, when none inequality is found so far, for each pair of clusters having vertices in S, say D_k and D_l, we check if the associated R-set-connectivity constraint is violated,

$$x(\delta(S)) < 2 \sum_{e \in E_R(S) \cap (V_k : V_l)} x_e.$$

If it certainly is, the reinforced R^+-set-connectivity inequality (16) is also violated. Otherwise, again an additional check is performed to see whether the reinforced D^+-set-connectivity inequality (16) is violated,

$$x(\delta_D(S)) + x(\delta_R(S) \setminus (V_k : V_l)) < 2 \sum_{e \in E_R(S) \cap (V_k : V_l)} x_e.$$

The procedure presented is exact for D-set-connectivity and R-set-connectivity constraints (13) and (15), but it is only heuristic for their reinforced versions (14) and (16).

6.2 Parity inequalities

Next, the separation methods for valid inequalities enforcing the parity of all vertices in the solution tour for the problem are shown. This involves the co-circuit and the matching constrains seen above.

6.2.1 Co-circuit inequalities

The exact method for separating co-circuit inequalities (17) in polynomial time follows the spirit of other exact separation procedures proposed for separating blossom inequalities [30].

For separating these inequalities some set S has to be found. As in the previous cases, it can be separated heuristically by finding the connected components in the solution graph G induced by edges with values $x_e > \varepsilon$, where ε is a given parameter. This leads to define candidate sets S, that once obtained, we need to find the odd subset of its cut, F. To achieve so, see that for a given set $S \subset V$ and a $F \subseteq \delta(S)$ with $|F|$ odd, the associated co-circuit inequality

$$x(\delta(S) \setminus F) - x(F) + |F| \geq 1,$$

might be rewritten as

$$\sum_{e \in \delta(S) \setminus F} x_e + \sum_{e \in F} (1 - x_e) \geq 1. \tag{24}$$

Thus, to solve the separation problem having a possible S, we must find some F in its cut that minimize the left hand side of (24) relative to a given vector x.

Note that the contribution of an edge $e \in \delta(S)$ to the left hand side of (24) is either x_e when $e \in \delta(S) \setminus F$ or $1 - x_e$ when $e \in F$. Then, for a given S, the smallest possible value of the left hand side of (24) is obtained for the set $F = \{e \in \delta(S) \mid 1 - x_e \leq x_e\} = \{e \in \delta(S) \mid x_e \geq 0.5\}$. When F defined this way is not odd, the smallest increment in the left hand side that guarantees that F is odd is obtained either by removing from or adding to one edge. Therefore, the smallest increase obtained is $\min\{\min\{x_e \mid e \in \delta(S) \setminus F\}, \min\{1 - x_e \mid e \in F\}\}$.

Therefore, the separation method for inequalities (24) consists of identifying the set S such that $\delta(S)$ contains the best possible set F. This is solved by computing the tree of mincuts of G, relative to the capacities vector given by x_e if $x_e < 0.5$ and by $1 - x_e$ otherwise.

Once the tree is obtained, we evaluate all its mincuts, since the smallest value of the left hand side of inequality (24) after making F odd is not necessarily associated with the smallest mincut of the tree.

6.2.2 Generalized co-circuit inequalities

We use a heuristic to separate violated generalized co-circuit inequalities (18), via their associated generalized matching inequalities (20) as indicated by Theorem 5.4 (see subsection 6.3.2 below).

6.3 Matching type inequalities

Firstly, the separation procedure for the simple matching inequalities (19) is detailed. And after that, we proceed by describing the method for the generalized ones, (20).

6.3.1 Separation of matching inequalities

In general, for a given solution x with $x(\delta(u)) = 1$ for all $u \in V$, matching inequalities $x(E(S)) \leq \frac{|S|-1}{2}$ are equivalent to inequalities $x(\delta(S)) \geq 1$, being $S \subset V$, with $|S|$ odd. It is well-known that the separation problem for this later expression can be solved with the algorithm of Padberg-Rao [31], which finds a minimum odd cut-set with respect to the capacities vector x. Nevertheless, these two inequalities are not equivalent when exists $x(\delta(u)) < 1$ for some $u \in V$.

In the case of the GARP, matching inequalities (19) only concern non-demand edges, but in G_R, the subgraph of the solution graph G induced by non-demand edges, the condition $x(\delta(u)) = 1$ does not necessarily hold for all u. An example has been shown in Figure 6, on page 14.

However, we can transform x and G_R into an equivalent vector and induced graph where all vertices have degree 1. Then the violated inequalities (19) are equivalent to violated odd cut-set inequalities, and thus can be separated with the Padberg-Rao algorithm.

In order to build this transformation, the connected components of G_R must be computed firstly. Let $H_t(V_t, E_t)$ denote these components, for $t \in T$. Observe that without loss of generality we can restrict the search of violated inequalities (19) to those associated with sets $S \subseteq V_t$. Then, for each connected component H_t the following procedure is performed:

1. If $x(E_t(V_t)) > \frac{|V_t|-1}{2}$, then the inequality (19) associated with $S = V_t$ is violated.
2. Else, we check whether the condition $x(\delta(u)) = 1$ holds for all $u \in V_t$ in H_t. If it does, then odd cut-set inequalities and inequalities (19) are equivalent in H_t. Thus, the outcome of the Padberg-Rao algorithm applied to H_t with the capacities vector inherited from x indicates whether a violated inequality (19) exists for some $S \subset V_t$.

   When $x(\delta(u)) < 1$ for some $u \in V_t$, then inequalities $x(\delta(S)) < 1$ and $x(E_R(S)) > \frac{|S|-1}{2}$ are not equivalent in H_t. In this case we define an "extended" component H'_t where these two inequalities are equivalent for all $S \subseteq V_t$. The set of vertices is $V'_t = V_t \cup \{w^t\}$, where w^t is a new vertex. Then, for all $u \in V_t$ with $x(\delta(u)) < 1$ we define a new edge (u, w^t) of capacity $x^t_{uw^t} = 1 - x(\delta(u))$. All other previous edges inherit their capacity, i.e. $x^t_e = x_e$ for all $e \in E_t$. Note that $x^t(E(S)) = x(E(S))$, for $S \subseteq V_t$. The outcome of the Padberg-Rao algorithm applied to the extended component indicates if $x^t(\delta(S)) < 1$ for some $S \subseteq V_t$. Observe that if a violated odd cut-set inequality exists for S with $w^t \in S$, the inequality is also violated for $S' = (V_t \cup \{w^t\})$. Since $x^t(\delta(u)) = 1$ for all $u \in V_t$, the outcome of the Padberg-Rao algorithm applied to the extended component also indicates if $x(E_R(S)) = x^t(E(S)) > \frac{|S|-1}{2}$ for some $S \subseteq V_t$.

Again, the computational burden of this exact procedure can be reduced by applying it onto the connected components of the subgraph of G_R induced by edges $e \in E_R$ with values $x_e > \varepsilon$ where ε is a given parameter.

6.3.2 Generalized matching inequalities

We apply a heuristic which reduces the separation of generalized matching inequalities (20) in the solution graph $G(V, E)$ to the separation of matching inequalities (19) in a graph of smaller size, G', in which some vertices and edges have been merged. Only pairs of vertices in the same component such that $(u, v) \notin E$ can be merged. This means that a pair $u, v \in V_k$ for some $k \in K$, is candidate for merging if either $(u, v) \in D_k$ and $x_{uv} = 0$, or no edge connecting u and v exists in the original graph.

A candidate pair or vertices is selected for merging if, either a third vertex i exists such that $x_{ui} > 0$ and

$x_{vi} > 0$, or another pair of vertices of the same cluster out of D_k, say $i, j \in V_l$, $l \neq k$ exist such that $x_{ui} > 0$ and $x_{vj} > 0$.

Merging two vertices i, j consists of replacing vertices i and j by one single vertex l and assigning to it the union of the cuts, $\delta(l) = \delta(i) \cup \delta(j)$. When merging these cuts, if i belongs to the same cluster as u and v, both edges (i, u) and $(j, u) \in D_k$. Therefore, the merged edge (l, u) has both end-vertices in the same cluster and is declared as a demand edge. In contrast, when u belongs to a different cluster from that of i and j, both edges (i, u) and (j, u) are non-demand. Therefore, the merged edge (l, u) (which connects the same pair of clusters as both (i, u) and (u, j)) is declared as non-demand.

The heuristic sequentially explores all clusters. Within each cluster it merges all pairs of candidate vertices satisfying any of the two criteria above. It is repeated again with the resulting shrunk graph, until no more vertices can be merged. Let $\overline{x}$ and $\overline{G} = (\overline{V}, \overline{E})$ respectively denote the weights vector and shrunk graph at the end of the process. Let also $\overline{V}_k$, $k \in K$ denote the shrunk vertex sets of the clusters. For $S \subset V$, its shrunk vertex set is denoted by $\overline{S} \subset \overline{V}$. For $\overline{u} \in \overline{V}$, its "original" vertex set is denoted by $S_{\overline{u}} \subset V$.

Observe that $\overline{x}(\overline{V}_k : \overline{V}_{k'}) = x(V_k : V_{k'})$, for all $k, k' \in K$, $k \neq k'$. Note also that, for $\overline{u} \in \overline{V}$, it holds that $x(E_D(S_{\overline{u}})) = 0$, since only pairs $(i, j) \notin E(x)$ can be merged. Therefore, we have

Theorem 6.1 *Let $\overline{S} \subset \overline{V}$ be the vertex set of a matching inequality* (19) *violated by $\overline{x}$ in $\overline{G}$. Then, the GMI* (20) *associated with S is violated by x in G.*

Let $\overline{S} = \{\overline{u}_1, \ldots, \overline{u}_r\} \subset \overline{V}$ with $h(\overline{u}_i) \neq h(\overline{u}_j)$, for $i \neq j$, i.e. in different original clusters. For $i = 1, \ldots, r$, let also $F_i = \delta_D(S^{\overline{u}_i})$. By Theorem 5.4 we also have the following corollary.

Corollary 6.1 *If the matching inequality* (19) *associated with $\overline{S}$ is violated by $\overline{x}$ in $\overline{G}$, the GCI* (18) *associated with $F = F_1 \cup \cdots \cup F_r$ is violated by x in G .*

7 A solution algorithm for the GARP

In this section we present a solution algorithm for solving the GARP using formulation F_x. It has two phases: the first one is an iterative LP based cutting plane algorithm, which starting from a relaxed formulation reinforces at each iteration the current formulation by adding valid inequalities violated by the current LP solution. The second phase is only applied when a provable optimal solution has not been found in the first phase, and resorts to CPLEX for solving exactly formulation F_x. For the first phase initially we consider the relaxation of formulation F_x which includes constraints (8) and (9), implying thus the connectivity constraints $x(\delta(V_k)) = 2$ for each cluster $k \in K$, but does not include any connectivity constraint (10). This formulation is further enhanced with the family of inequalities

$$x(V_k : V_l) \leq 1 \qquad k, l \in K, k \neq l \tag{25}$$

which are valid for any alternating tour.

As before, let $G(V, E)$ denote the support graph associated with the solution x at any iteration of the cutting plane algorithm. The strategy we use for finding violated inequalities is the following (using $\varepsilon \in \{0, 0.05, 0.1\}$ in all cases):

Step 1: Cluster-connectivity inequalities of type (10) for $S = \cup_{k \in K_S} V_k, \quad K_S \subseteq K$ (see Section 6.1). We operate on the graph G_k with its weighted edges as capacities. We first identify the connected components in the graph induced by edges with $x_e > \varepsilon$. Then, if no violated inequality (10) has been found, we apply the exact separation procedure by finding the tree of min-cuts relative to the capacities vector.

Step 2: General case of cluster-connectivity inequalities (10) and reinforced set-connectivity inequalities (14) and (16) (see Section 6.1.1). We proceed as in Step 1, with the only difference that we work on the graph G relative to the capacities given by x. For each candidate set S, we check whether $V_k \subseteq S$ for some $k \in K$ to find out the type of inequality it may produce: a violated inequality (10) in the former case or set-connectivity in the latter one.

Step 3: Co-circuit inequalities (17). We proceed as indicated in Section 6.2. We apply the heuristic first, and only if it fails we apply the exact separation.

Step 4: Matching inequalities (19). We proceed as indicated in Section 6.3.1. As before, we apply the heuristic first, and only apply the exact separation when it fails.

Step 5: General matching inequalities (20). We proceed as indicated in Section 6.3.2.

A violated inequality of any kind is added to the current LP only if its violation is greater than or equal to a given parameter that we have fixed at 0.1.

When violated inequalities are no longer found and the current LP solution is not integer we apply the second phase in which CPLEX is used for solving exactly formulation F_x reinforced with the violated inequalities found in the first phase. Inequalities (8) guarantee that integer solutions to the current formulation satisfy the parity constraints. However, it is possible that integer solutions to the current reinforced formulation do not satisfy all connectivity constraints (10) associated with sets S which are the union of vertex sets. For this reason, in the second phase we use as callback function for CPLEX a separation routine for such connectivity constraints (the same that is used in Step 1), which is only applied at the nodes of the search tree where an integer solution is found. In this way the second phase terminates with an integer solution satisfying all connectivity constraints (10) which is optimal for formulation F_x.

8 Computational experiments

Next we describe the computational experiments we have run and we report on the results obtained. The programs were coded in IBM® ILOG® Script and run with the IBM ILOG CPLEX Optimization Studio Version 12.4. Default parameters were used. All the experiments were run on an Intel(R)Core(TM)2 Quad CPU Q9400 at 2.66GHz and 8 GB of RAM using a 64-bit Operating System.

Three sets of instances have been used in these computational experiments. The first two contain well-known instances used as benchmarks for numerous arc routing problems. As might be seen in the next section, for all these instances an optimal solution was already found without the need of the branch and cut algorithm. The cutting plane was enough to solve these instances in 677 seconds. For this reason a new set of instances was generated specially for the GARP. Thus, the third set contains some new larger instances. Later in this section, the steps to build these data will be described.

8.1 Instances already existent in the literature

The first two sets are:

S1 The 118 clustered prize-collecting arc routing problem (CPARP) instances used by [4] for the clustered prize-collecting arc routing problem and in [13] for the windy prize-collecting arc routing problem.

S2 The 40 general routing problem (GRP) instances of http://www.uv.es/corberan/instancias.htm used in [13] for the windy prize-collecting arc routing problem.

The 118 instances of the set S1 are divided into five groups. The first group contains two instances, ALBAIDAA and ALBAIDAB, [17]. The second group, instances labeled P, contains the 24 instances from [18]. The last three groups contain instances from [26]: 36 instances with vertices of degree 4 (labeled D), 36 grid instances (labeled G), and 20 randomly generated instances (labeled R).

The 40 instances of the group S2 are divided into three groups. Sets ALB and MAD contain 15 instances each, generated from the street networks of the Spanish towns of Albaida and Madrigueras, [11]. The set GRP contains 10 randomly generated General Routing Problem instances [14].

Like in other works where instances in sets S1 and S2 have been used in a clustered context, the clusters have been set to be the connected components induced by the required edges of the original RPP or GRP instance. In all cases, vertex 1 has been taken as depot.

8.2 Instances specially generated for this problem

All instances of the group S3 have 300 vertices. They where generated following the next steps:

1. Start with an Euclidean bidimensional space sizing 10000×10000 points.
2. Set the vertices at randomly chosen locations with minimum distance of 50 points between them.
3. For each vertex, add a random quantity of edges connecting it to others with distance under 500. These degrees follow a uniform distibution, $|\delta(v)| \sim U[6, 30]$. As costs, the integral part of their distances are taken.
4. Add edges to connect the distinct connected components obtained so far. These edges are created from a random quantity, t, of minimum spanning trees among these connected components. This quantity follows a uniform distribution $t \sim U[10, 50]$, too.
5. Purge the graph by deleting all edges uv if $c_{uv} < 0.98(c_{uw} + c_{wz})$ for some vertex w. This, in order to avoid almost-parallel edges.

At this point we have already created the graph. Continue to define the clusters.

6. Choose p centroids, one for each new cluster. Each centroid is a randomly selected vertex, but neighbours of any previously selected as centroid.
7. Assign the rest of vertices to their nearest centroid. This might lead to have more than p clusters since one node may not be connected to its nearest centroid.
8. Define as demand edge, each one already existent with both end-vertices in the same cluster.
9. Once at this point, for each node with $\delta_D(v) = \emptyset$ either:
    - If there is no edge joining the node with its nearest centroid, add a new edge between it and the nearest node already belonging to some cluster.

- If it is an isolated centroid, join it to the nearest cluster without creating new edges, but defining the corresponding edge as demand one. This might reduce the number of clusters.

By this mean, 10 instances with $p \simeq 50$ have been created, and 10 more with $p \simeq 30$.

In Table 1, information on all is depicted in groups according to their characteristics and sizes. Column under *Instances* gives the number of instances in the group, followed by the numbers of vertices n, edges m, demand edges $|D|$, and clusters, p. When all values do not coincide, minimum and maximum values in the group are given.

	Instances	n	m	$\|D\|$	p
ALBAIDA	2	90–102	144–160	88–99	10–11
P	24	7–50	10–184	4–78	2–8
D16	9	16	31–32	3–16	2–5
D36	9	36	72	10–38	5–12
D64	9	64	128	27–75	5–15
D100	9	100	200	50–121	9–23
G16	9	16	24	3–13	4–6
G36	9	36	60	11–35	5–10
G64	9	64	112	24–68	4–15
G100	9	100	180	41–113	4–21
R20	5	20	37-75	3–7	4–5
R30	5	30	70-112	7–11	5–7
R40	5	40	82-203	8–18	6–9
R50	5	50	130-203	13–20	6–12
ALBA_3	5	116	174	44–57	16–24
ALBA_5	5	116	174	88–92	9–18
ALBA_7	5	116	174	113–122	2–9
GRP	10	116	174	52–126	5–35
MADR_3	5	196	316	86–108	34–43
MADR_5	5	196	316	147–163	21–27
MADR_7	5	196	316	211–238	2–7
GARP50	10	300	43000–45000	447–522	46–52
GARP30	10	300	43000–45000	567–613	25–36

Table 1: Summary of instances.

8.3 Numerical results

Throughout, z_0 denotes the value of the solution to the initial LP formulation, z_r the lower bound obtained after the cutting plane algorithm, and z^* denotes the optimal integer value.

In Table 2 results for sets S1 and S2 are depicted. As already said, all instances of these sets where solved optimally with the cutting plane algorithm only. Thus, in Table 2, $z_r = z^*$. For this reason, columns show

the deviation from the solution to initial formulation with no inequalities added, to the final solution of the cutting plane algorithm, that is, the integer optimal solution for these instances. These deviations are printed in absolute, $z^* - z_0$, and relative terms, $(z^* - z_0)/z^*$. The average number of iterations and the average time for each group may be seen in last two columns.

	$z^* - z_0$		$(z^* - z_0)/z^*$		iterations	time (s)
	avg	max	avg	max		
ALBAIDA	1224.00	1592	0.29	0.37	75.50	8.338
P	5.08	32	0.05	0.24	5.13	0.222
D16	6.00	18	0.01	0.03	1.67	0.055
D32	40.56	100.5	0.08	0.19	11.78	0.420
D64	55.06	149	0.08	0.24	11.22	0.831
D100	126.58	209	0.15	0.26	51.33	5.905
G16	0.22	1	0.02	0.08	2.00	0.068
G32	1.00	3	0.05	0.17	5.22	0.237
G64	3.11	7	0.13	0.32	20.67	1.418
G100	4.81	11	0.14	0.32	43.44	4.984
R20	0.00	0	0.00	0.00	1.00	0.060
R30	193.40	967	0.01	0.03	2.00	0.124
R40	1392.70	5662	0.04	0.16	1.80	0.186
R50	2914.20	5439.5	0.08	0.12	9.60	0.451
ALBA3	735.40	606	0.15	0.18	41.40	3.341
ALBA5	936.27	1332	0.22	0.33	104.00	17.005
ALBA7	791.60	1948	0.18	0.43	75.60	7.417
GRP	955.70	1358	0.19	0.29	64.40	7.943
MADR3	411.67	372.5	0.06	0.09	37.60	17.553
MADR5	978.00	1427.5	0.19	0.26	60.40	29.996
MADR7	374.00	950	0.13	0.36	44.20	7.381

Table 2: Summary of results for sets S1 and S2.

Results for the larger instances of set S3 are shown in Table 3. The first column, with the ✓, gives the number of instances for which an optimal solution was found with the cutting plane algorithm. Next four columns give the average and maximum deviation of the solution to the initial formulation z_0 and the final lower bound z_r, from the optimal value z^* in relative terms. Follow the averages of the number of iterations carried out and the times in seconds spent in the LP, t_{z_r}. After that, averages of nodes explored in the search tree by the exact algorithm and the total times, t_{z^*} also in seconds, are shown.

	✓	$(z^* - z_0)/z^*$		$(z^* - z_r)/z^*$		iterations	t_{z_r} (s)	nodes	t_{z^*} (s)
		avg	max	avg	max				
GARP50	1	0.1249	0.1744	0.0000	0.0000	245	4196.045	391	4460,388
GARP30	2	0.1957	0.3358	0.0090	0.0247	366	7420.331	790	8673.278

Table 3: Summary of results for set S3.

In group GARP50 just one was solved without the branch and cut procedure. However, all nine other reached the optimal value as the lower bound. Thus, for these cases the task done by the branch and cut part was getting the final integer values for the variables. On the other hand, note also that the time to solve the ten instances of GARP30 set required almost one whole day.

Nevertheless, the effectiveness of the cuts can be appreciated by comparing the values in the entries of columns $(z^* - z_0)/z^*$ and $(z^* - z_r)/z^*$, which illustrate the reduction in the deviation of the lower bound, with respect to the optimal value, initially and at termination of the cutting plane algorithm.

The number of violated inequalities found by the separation procedures are summarized in Table 4.

	connectivity		co-circuit		matching	
	heuristic	exact	heuristic	exact	heuristic	extended
ALBAIDA	20	424	–	–	–	–
P	34	115	–	–	–	–
D16	–	8	–	–	–	–
D36	13	122	–	5	–	–
D64	14	129	–	–	–	–
D100	58	941	20	80	2	1
G16	1	5	4	2	1	1
G36	5	52	8	2	–	–
G64	22	404	–	–	–	–
G100	45	962	16	26	2	1
R20	–	–	–	–	–	–
R30	1	4	–	–	–	–
R40	2	4	–	–	–	–
R50	5	50	–	8	–	1
ALBA3	37	424	4	–	–	–
ALBA5	56	1219	–	–	–	–
ALBA7	6	615	–	–	–	–
GRP	79	1105	16	11	–	2
MADR3	32	379	20	2	–	–
MADR5	38	1021	–	–	–	–
MADR7	21	332	–	–	–	–
GARP50	283	9894	80	206	–	4
GARP30	409	21194	10	30	–	–

Table 4: Total number of different types of inequalities.

Columns under *connectivity* indicate the number of connectivity inequalities separated with the heuristic and with the exact algorithm. The next two columns, *co-circuit* give the number of co-circuit inequalities separated by heuristic methods and by the exact one. Heuristic methods include the one used in Aráoz, Fernández and Franquesa [4], and the procedures that work on the shrunk graph induced by the solution. The columns under *matching* show matching inequalities separated by the heuristic, and the inequalities separated when considering the generalized matching inequalities heuristically. As can be seen in Table 3, most of inequalities added during the iterative procedure were connectivity inequalities, although some of the other types were also used.

Acknowledgements

This research has been partially supported by the Spanish Ministry of Economy and Competitiveness and EDRF funds through grant MTM2015-63779-R (MINECO/FEDER). This support is gratefully acknowledged.

References

[1] C. C. Orloff (1974). A Fundamental Problem in Vehicle Routing. *Networks.* 4, 35-64.

[2] A. Corberán, G. Laporte (2014). Arc Routing: Problems, Methods, and Applications. *SIAM.*

[3] Ahr, D. (2004). Contributions to Multiple Postmen Problems. PhD dissertation, Departament of Computer Science, Heidelberg University, Heidelberg, Germany.

[4] Aráoz, J., E. Fernández and C. Franquesa, (2009). The Clustered Prize-collecting Arc-routing Problem, *Transportation Science* 43, 287–300.

[5] Aráoz, J., E. Fernández and C.Franquesa, C., The generalized arc routing problem, ROUTE 2011, Sitges, June 2011.

[6] Aráoz, J., E. Fernández and C. Franquesa, C., The generalized arc routing problem, Conference of the International Federation of Operational Research Societies (IFORS 2011). Melborune. July 2011.

[7] Aráoz, J., E. Fernández, O. Meza. (2009). An LP-based algorithm for the privatized rural postman problem. *European Journal of Operational Research* 196(3) 886–896.

[8] Ávila, T., Á. Corberán, I. Plana, J.M. Sanchis, (2015). A new branch-and-cut algorithm for the generalized directed rural postman problem, *Transportation Science*, http://dx.doi.org/10.1287/trsc.2015.0588.

[9] Barahona, F. and M. Grötschel, (1986). On the cycle polytope of a binary matroid. *J. Comb. Theory* 40 40–62.

[10] Belenguer, J. M. and E. Benavent, (1998). The capacitated arc routing problem: Valid inequalities and facets. *Comput. Optim. Appl.* 10, 165–187.

[11] Benavent, E., A. Carrotta, A. Corberán, J.M. Sanchis, and D. Vigo (2007). Lower bounds and heuristics for the windy rural postman problem. *Eur J Oper. Res.* 176, 855–869.

[12] Benavent, E., A. Corberán and J.M. Sanchis (2000). Linear Programming Based Methods For Solving Arc Routing Problems. In *Arc Routing: Theory, Solutions and Applications.* M. Dror (ed.) Kluwer Academic Publishers, 2000. 231–275.

[13] Corberán, A., E. Fernández, C. Franquesa, and J.M. Sanchis (2011). The windy clustered prize-collecting arc routing problem. *Trans. Sci.* 45, 317–344.

[14] Corberán, A., A. Letchford, and J.M. Sanchis (2001). A cutting plane algorithm for the general routing problem. *Math. Program.* 90, 291–316.

[15] Corberán, A., Plana, I., Rodríguez-Chía, A., Sanchis, J.M., (2011). A Branch-and-Cut for the Maximum Benefit Chinese Postman Problem. *Math. Program.*, DOI: 10.1007/s10107-011-0507-6.

[16] Corberán, A., I. Plana and J.M. Sanchis (2015). Distance Constrained Generalized Directed Rural Postman Problem. Workshop on Combinatorial Optimization, Routing and Location, CORAL 2015. Salamanca (Spain), 30 Sep-2 Oct 2015.

[17] Corberán, A. and J. M. Sanchis. 1994. A polyhedral approach to the rural postman problem. *Eur. J. Oper. Res.* 79, 95–114.

[18] Christofides, N., V. Campos, A. Corberán, and E. Mota. 1981. An algorithm for the rural postman problem. *Imperial College Report IC.O.R.*, 81.5.

[19] Drexl, M., (2007). On some Generalized Routing Problems, Ph.D. thesis, Aachen University, Aachen, Germany.

[20] Drexl, M., (2014). On the Generalized Directed Rural Postman Problem, *The journal of the Operational Research Society* 65, 1143–1154.

[21] Edmonds, J. 1965. Maximum matchings and a polyhedron with 0–1 vertices. *Jornal of Research National Bureau of Standards* 69B, 125–130.

[22] Fernández, E., Some results on the Generalized Arc Routing Problem, 1st Workshop on Arc Routing Problems, Copenhagen, May 2013.

[23] Fernández, E., On the Generalized Arc Routing Problem, 8th Triennial Symposium on Transportation Analysis (TRISTAN VIII), Atacama, June 2013.

[24] Ghiani, G. and G. Laporte (2000). A branch-and-cut algorithm for the undirected rural postman problem. *Math. Program.* 87, 467–481.

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

[26] Hertz, A., G. Laporte and P. Nanchen-Hugo (1999). Improvement procedures for the undirected rural postman problem. *INFORMS J. Comput.* 1, 53–62.

[27] Laporte, G. and Y. Nobert (1983). Generalized traveling salesman problem through n Sets of nodes: An integer programming approach. *INFOR* 21, 61–75.

[28] Laporte, G.. H. Mercure, and Y. Nobert (1987). Generalized Traveling Salesman Problem Through n Sets of nodes: The Asymmetrical Cases. *Discrete Applied Mathemtics* 18, 185–197.

[29] Laporte, G., A. Asef-Vaziri, and C. Sriskandarajah (1996). Some Applications of the Generalized Traveling Salesman Problem. *The Journal of the Operational Research Society* 47, 1461–1467.

[30] Letchford, A.N., G. Reinelt and D.O. Theis, (2008). Odd minimum cut-sets and b-matchings revisited. *SIAM J. Discrete Math.* 22, 1480–1487.

[31] Padberg, M. and M.R. Rao, (1982). Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research* 7, 67–80.

[32] R. Shuttleworth, Golden, B.L., S. Smith and E. Wasil, (2008). Advances in meter reading: Heuristic solution of the close enough traveling salesman problem over a street network, in The Vehicle Routing Problem: Latest Advances and New Challenges, B.L. Golden, S. Raghavan, and E. Wasil, eds. Springer, Berlin, pp. 487–501.