A Branch-and-Cut Algorithm for the Multi-Depot Rural Postman Problem

Elena Fernández [*1, 2], Gilbert Laporte [†3], and Jessica Rodríguez-Pereira [‡1]

[1]Department of Statistics and Operations Research, Universitat Politècnica de Catalunya-BcnTech, Spain
[2]Barcelona Graduate School of Mathematics (BGSMath)
[3]Canada Research Chair in Distribution Management, HEC Montréal, 3000, chemin de la Côte-Sainte-Catherine, Montreal H3T 2A7, Canada

May 15, 2017

Abstract

This paper considers the Multi-Depot Rural Postman Problem, an extension of the classical Rural Postman Problem in which there are several depots instead of only one. The aim is to construct a minimum cost set of routes traversing each required edge of the graph, where each route starts and ends at the same depot. The paper makes the following scientific contributions: *i)* it presents optimality conditions and a worst-case analysis for the problem; *ii)* it proposes a compact integer linear programming formulation containing only binary variables, as well as a polyhedral analysis; *iii)* it develops a branch-and-cut algorithm that includes several new exact and heuristic separation procedures. Instances involving up to four depots, 744 vertices, and 1315 edges are solved to optimality. These instances contain up to 140 required components and 1000 required edges.

Key words: Arc routing; multi-depot rural postman problem; worst-case analysis; polyhedral analysis; branch-and-cut.

1 Introduction

The purpose of this paper is to introduce a compact model and to develop a branch-and-cut algorithm for the Multi-Depot Rural Postman Problem (MDRPP), which extends the classical Rural Postman Problem (RPP) [30] where there is only one depot, but in the MDRPP there are several. Similarly to the RPP, routes must be designed to serve a given set of required edges. In contrast, in the MDRPP

*e.fernandez@upc.edu
†gilbert.laporte@cirrelt.ca
‡jessica.rodriguez@upc.edu

the depot from which each required edge is served is not known in advance. The MDRPP combines two types of decisions: the allocation of required edges to depots and the planning of routes. The objective is to determine a minimum cost set of routes, each starting and ending at the same depot, and such that each required edge is traversed at least once.

The motivation for studying the MDRPP comes not only from its theoretical interest but also from its real-life applications. Similarly to other arc routing problems, such applications arise in a wide variety of practical cases, namely garbage collection, road maintenance, mail delivery, snow plowing or pipelines inspection, to name just a few (see e.g. [9]). In large-scale instances, there is usually more than one depot from which service demand can be satisfied. Such depots may be vehicle stations, dump sites, replenishment points or relay boxes. A way of handling such problems is to first define a smaller operating area for each depot, by using a districting procedure in which each district contains a single depot, and then solving the RPP associated with each district. This solution strategy is of course suboptimal.

The literature on Multi-Depot Arc Routing Problems (MDARP) is scarce. To the best of our knowledge, [14, 16] present the only existing exact algorithms for the MDRPP. Both use *natural* decision variables which explicitly indicate the depot to which each traversed edge or arc is associated. An exact branch-and-cut based on a binary linear formulation was proposed in [16] for the MDRPP on an undirected graph. A directed MDRRP is considered in [14], where an exact branch-and-cut algorithm is developed for a collaborative arc routing problem. In [16] which deals with an undirected MDRPP, instances with up to 100 vertices and four depots are solved to optimality. Since the lengths of optimal routes in the MDRPP may be very unbalanced, this paper also considers a min-max type objective function aiming at balancing the length of the routes. In [14], which addresses a directed MDARP dealing with carriers collaboration, instances with up to 50 vertices and two depots are optimally solved. Other than this, previous work on MDARPs has focused on multi-depot capacitated arc routing problems (MDCARPs). Some theoretical aspects of MDCARPs are considered in [31]. A new formulation and exact solution algorithm are presented in [25] for the asymmetric multi-depot capacitated arc routing problem. Heuristics have been put forward for both the undirected and the directed MDCARPs. Sequential heuristics for the undirected MDCARP are proposed in [1, 27, 28]. A cluster-first-route-second strategy, where the assignment of arcs to depots is established before designing the routes is applied in [1], and a route-first-cluster-second strategy is used in [27, 28], where a single giant route is created first and later partitioned into smaller routes. Population based heuristics have also been used for solving MDCARPs. For the undirected case, two different ant colony strategies are presented in [24], and a hybrid genetic algorithm with perturbation that incorporates a local search, a replacement method, and a perturbation mechanism is proposed in [23]. The directed case is addressed in [32], where an evolutionary approach is presented, which takes advantage of the extensions of the heuristics for the classical single-depot Capacitated Arc Routing Problem [19].

Multi-depot routing problems are also related to districting-arc routing problems where a set of clusters or districts that suitably partition the required edge set is sought. The design of good districts takes place at a strategic level, where demand points or edges are allocated to depots, and allows finding efficient routes in each district at an operational level in a later phase. There exists a rich districting literature in relation to arc routing. In fact, some of the above referenced works stem from this research area. As an example, the heuristics of [27, 28] are devised as a second phase in districting problems. Two recent works on districting for arc routing are [6, 17]. The interested reader is referred to [26, 29] for further readings on this topic.

This paper makes the following scientific contributions: *i*) it presents optimality conditions and a worst-case analysis for the MDRPP; *ii*) it proposes a compact integer linear programming formulation containing only binary variables for the problem, as well as a polyhedral analysis; *iii*) it develops a branch-and-cut algorithm that includes several new exact and heuristic separation procedures. We show that instances involving up to four depots, 744 vertices, and 1315 edges can be solved to optimality. These instances contain up to 140 required components and 1000 required edges.

The remainder of the paper is organized as follows. Section 2 contains a formal definition of the problem, as well as some properties. The mathematical model and the polyhedral analysis are presented in Section 3, followed by the branch-and-cut algorithm in Section 4. Extensive computational results are presented in Section 5. The paper closes with some conclusions in Section 6.

2 Formal definition and properties

The MDRPP is defined on an undirected connected graph $G = (V, E)$, where V is the vertex set, $|V| = n$, E is the edge set, and $|E| = m$. We denote by $D \subset V$ the set of depots, by $R \subset E$ the set of required edges, and by $F = E \setminus R$ the set of unrequired edges. The connected components induced by the required edges are referred to as *required components* and indexed in a set K. These components are denoted by $C_k = (V_k, R_k)$, $k \in K$, so $R = \bigcup_{k \in K} R_k$. Let $V_R = \bigcup_{k \in K} V_k$. We assume that E contains no edge connecting two depots, and that no component has more than one depot, although it is possible that a component contains no depot, i.e. $|V_k \cap D| \leq 1$ for all $k \in K$. Let c be a non-negative real cost function defined on the edges of G.

In the following we assume that G has been simplified so that V is the set of vertices incident to the edges of R, and E contains the edges of R plus additional unrequired edges, connecting every pair of vertices not connected with an edge of R, representing shortest paths in the original graph. To this end, following the procedure described in [7], we first add to $G_R = (V_R, R)$ an edge between every pair of vertices of V_R having a cost equal to the shortest path length on G. Hence the costs of the simplified graph satisfy the triangle inequality.

We use the term *route* to denote a closed but not necessarily simple path that starts and ends at the same depot $d \in D$. We say that a required edge $e \in R$ is *served* if a route traverses it at least once. As usual, the cost of a route is the sum of the costs of the edges in the route, where the cost of each edge is counted as many times as it is traversed.

We denote by T_C the Minimum Spanning Tree (MST) with respect to cost function c, of the multigraph $G_C = (V_C, E_C)$ induced by the connected components. In addition, we will use the following usual notation. For any non-empty vertex subset $S \subset V$, $\delta(S) = \{(u, v) \in E | u \in S, v \in V \setminus S\} = \delta(V \setminus S)$ is the set edges in the cut between S and $V \backslash S$ and $\gamma(S) = \{(u, v) \in E | u, v \in S\}$ the set of edges with both vertices in S. For a singleton $S = \{v\}$, with $v \in V$, we just write $\delta(v)$ instead of $\delta(\{v\})$. For $H \subset E$ we use $\delta_H(S) = \delta(S) \cap H$ and $\gamma_H(S) = \gamma(S) \cap H$. Furthermore, a vertex $v \in V$ is H-odd if $|\delta_H(v)|$ is odd; otherwise v is H-even. Finally, we use the standard compact notation $f(A) \equiv \sum_{e \in A} f_e$ where $A \subseteq E$, and f is a vector or a function defined on E. If f is only defined on subset $B \subset E$, we use $f(A) \equiv f(A \cap B) \equiv \sum_{e \in A \cap B} f_e$.

In the remainder of this paper, we make the following modeling assumption:

H1) Required edges in the same component can be served in routes from different depots.

Figure 1 illustrates the effect of this assumption. Figure 1a shows the input graph, which has two required components with a depot in each one of them (v_1 and v_2, respectively). The black lines represent required edges, while the unrequired edges are drawn in light grey. The numbers next to the edges indicate their cost. Figure 1b shows the optimal solution of cost $z = 23$ when we impose that all required edges in the same component be served from the same depot. The route of depot v_1 (represented with solid lines), which serves the required edges of C_1, consists of edges (v_1, A), (A, B), and (B, v_1). The route of depot v_2 (represented with dotted lines), which serves the required edges of C_2, consists of edges (v_2, E), (E, C), (C, D), (D, E), (E, F), and (F, v_2). Figure 1c shows that a better solution of cost $z = 19$ can be obtained if we allow to serve required edges in the same component from different depots. Now all the required edges of C_1 and some required edges of C_2 are served in the route from depot v_1 defined by edges (v_1, A), (A, C), (C, E), (E, D), (D, B), and (B, v_1). The remaining required edges of this component are served in the route from depot v_2, which consists of edges (v_2, E), (E, F), and (F, v_2).

We note that, as a consequence of the modeling assumption H1, feasible routes are not necessarily vertex-disjoint.

The MDRPP is to find a set of non-empty routes, one from each depot, that serve all the required edges at minimum total cost.

2.1 Optimality conditions

As is usual in related uncapacitated arc routing problems on undirected graphs, the MDRPP satisfies some optimality conditions that allow to derive formulations using binary variables only (see, for instance, [15, 18]). In particular, the following properties were proven in [16]:

O1) There exists an optimal MDRPP solution in which each required edge is served by exactly one route.

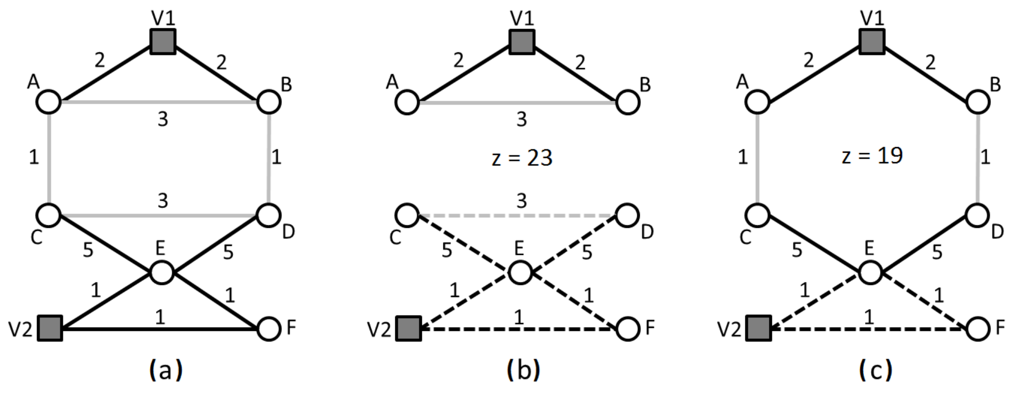


Figure 1: Example showing that allowing to split the required components among routes may produce better solutions

O2) There exists an optimal MDRPP solution in which no edge is traversed more than twice.

O3) There exists an optimal MDRPP solution where no unrequired edge with the two end-vertices in the same component ($e \in \gamma_F(V_k)$) is traversed more than once. Furthermore, because of the triangle inequality, the only edges of $\gamma_F(V_k)$ that are used connect two R-odd vertices.

O4) There exists an optimal MDRPP solution in which the only unrequired edges that are traversed twice are the edges of T_C. (See [18] for a similar result on the RPP).

2.2 Worst-case analysis

In this section we make a worst-case comparison between the MDRPP and the RPP. We close the section with an analysis of the improvement that can be obtained due to the modeling hypothesis H1 that allows serving the edges of a required component from different depots. Throughout the section we denote by $z^*(MDRPP)$ the optimal value of an MDRPP instance and by $z^*(RPP)$ the optimal value of the same instance with only one depot. Note that, when all depots are incident to a required edge, the optimal value $z^*(RPP)$ of an RPP instance on a given graph is independent of the location of the depot. We will also use the notation $z(H)$ to indicate the total cost of the edges in $H \subset E$, for an MDRPP or an RPP instance.

The costs savings that can be obtained with the MDRPP with respect to the RPP with one single depot can be arbitrarily large. The highest savings are achieved when a depot is located in each component. Then $z^*(MDRPP)$ is the sum of the optimal Chinese Postman solution values on each component. Figure 2 illustrates one such example, which we will use in the proof of Theorem 2.1.

Theorem 2.1 *There exists no finite bound for the ratio $z^*(RPP)/z^*(MDRPP)$.*

Proof: Consider an MDRPP instance like the one depicted in Figure 2, defined on

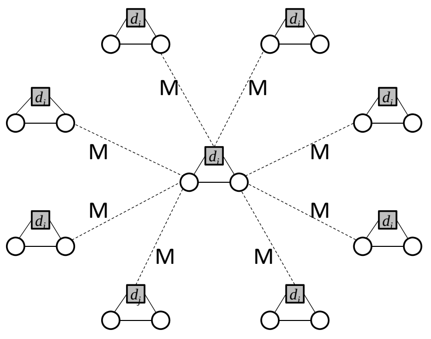


Figure 2: Potential improvement of the MDRPP relative to the RPP

a graph $G = (V, E)$, where each required component C_k, $k \in K$, contains a depot, represented by a grey square, and its required set R_k consists of a triangle (solid lines). One required component is located at the center of an imaginary circle and the remaining $|K| - 1$ components are displayed around its circumference. The set F of unrequired edges are $|K| - 1$ radii of the circle, each of them connecting the center component and one of the other components (dotted lines). The cost of each unrequired edge is M. It is clear that the optimal value of the MDRPP is the sum $z(R)$ of the costs of all required edges. It is also easy to see that the cost of the RPP with only one depot is $z^*(RPP) = z(R) + 2z(F) = z(R) + 2\,(|K| - 1)\,M$. Therefore

$$\frac{z^*(RPP)}{z^*(MDRPP)} = \frac{z(R) + 2\,(|K| - 1)\,M}{z(R)},$$

which tends to ∞ when $M \to \infty$.

Despite the above result, it is also possible that $z^*(MDRPP)$ will be higher than $z^*(RPP)$. Broadly speaking, this will happen when the need of using all the depots worsens the potential quality of a solution. Below we give a lower bound on the ratio $z^*(RPP)/z^*(MDRPP)$.

Theorem 2.2 $z^*(RPP)/z^*(MDRPP) \geq 1/2$*, and the bound is asymptotically tight.*

Proof: To see that $z^*(MDRPP) \leq 2z^*(RPP)$ we observe that a feasible solution for a given MDRPP instance can be obtained from an optimal RPP solution as follows:

i) Replicate all the edges of the RPP solution.

ii) Eliminate all pairs of unrequired edges connecting two components, both containing one depot.

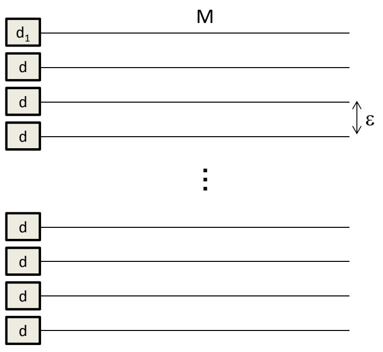
iii) For each component containing no depot, retain one dipath connecting it with some component with a depot, and eliminate all remaining such dipaths if they exist.

The cost of the solution after *i)* is $2z^*(RPP)$. Thus if z^* denotes the cost of the feasible MDRPP solution at the end of the process we have $z^* \leq 2z(RPP)$. Since the optimal MDRPP value cannot be greater than z^* we have $z^*(MDRPP) \leq z^* \leq 2z^*(RPP)$. To see that the bound can be attained asymptotically we provide an example.

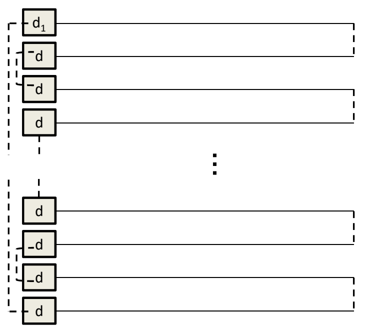
Consider an MDRPP instance like the one depicted in Figure 3a defined on a graph $G = (V, E)$, where each required component consists of one single edge represented by a solid line, $R_k = \{(u_k, v_k)\}$ of cost M, and contains a depot (gray square) located at its leftmost vertex u_k. Suppose that all required edges are parallel. The set of unrequired edges (dotted lines) contains edges connecting the leftmost and rightmost end-vertices of each consecutive pair of edges, i.e. $F = \{(u_k, u_{k+1}) : 1 \leq k < |K|\} \cup \{(v_k, v_{k+1}) : 1 \leq k < |K|\}$. Let us finally suppose that the cost of each unrequired edge is ε. In the optimal MDRPP solution to the above instance, each required edge is traversed twice and no unrequired edge is used. Hence, $z^*(MDRPP) = 2|K|M$. The optimal RPP solution (see Figure 3b) traverses each required edge only once and connects each pair of consecutive components with one small unrequired edge (in total $|K| - 1$ such edges) of cost ε. Finally, the last and first components are connected with a path of unrequired edges, which traverses all the components (in total $|K| - 1$ small edges again). The value of the optimal RPP solution is thus $z^*(RPP) = |K|M + 2(|K| - 1)\varepsilon$. Therefore, for the instance described above we have

$$\frac{z^*(RPP)}{z^*(MDRPP)} = \frac{|K|M + 2(|K|-1)\varepsilon}{2|K|M},$$

which tends to 1/2 when $\varepsilon \to 0$. ■



(a) Input graph



(b) Optimal RPP solution

Figure 3: Potential improvement of the RPP relative to the MDRPP.

We conclude this section by comparing the value of the MDRPP (in which, according to H1, the edges of a required component can be served from different depots), with its clustered version MDRPP$_C$ (in which it is imposed that all the required edges in the same component are served from the same depot). Denote by $z^*(MDRPP_C)$ the optimal value to a given MDRPP$_C$ instance. Since any feasible solution to the MDRPP$_C$ is feasible for the MDRPP, we have that $z^*(MDRPP) \leq z^*(MDRPP_C)$. Thus a lower bound for the ratio $z^*(MDRPP_C)/z^*(MDRPP)$ is one. It is easy to construct examples for which both problems have the same optimal solution, so that this lower bound is tight. Below we give a result on an upper bound of the ratio $z^*(MDRPP_C)/z^*(MDRPP)$ and show that the bound is asymptotically tight.

Theorem 2.3 *$z^*(MDRPP_C)/z^*(MDRPP) \leq 2$, and the bound is asymptotically tight.*

Proof: To see that $z^*(MDRPP_C) \leq 2z^*(MDRPP)$ it is sufficient to observe that replicating all the edges in an optimal MDRPP solution yields a solution in which the edges of each required component define an Eulerian graph, and all the edges connecting two different components, are used an even number of times. It is thus sufficient to remove two copies of some of the edges connecting two different components to obtain a feasible solution to the MDRPP$_C$.

The example of Figure 4a shows that the bound is asymptotically tight. Consider a graph with an even number of required components, where each required component consists of two edges depicted with solid lines: a small one, (u_k, v_k) of cost δ, and a long one, (v_k, w_k) of cost M. Suppose that the required edges in each component are aligned and that all required components are parallel. Each component contains a depot represented with a light gray square. The depot of component one is located at its rightmost vertex w_1, whereas the depots of all other components are located at their leftmost vertex u_k. The unrequired edges are shown by dotted lines. They connect pairs of *similar* vertices in consecutive components, i.e. $F = \{(u_k, u_{k+1}) : 1 \leq k < |K|\} \cup \{(v_k, v_{k+1}) : 1 \leq k < |K|\} \cup \{(w_k, w_{k+1}) : 1 \leq k < |K|\}$. The cost of each unrequired edge is ε.

An optimal solution to MDRPP$_C$ is obtained replicating all required edges, since all the edges of each required component must be served from its depot. The value of this solution is $z^*(MDRPP_C) = 2|K|(\delta + M)$.

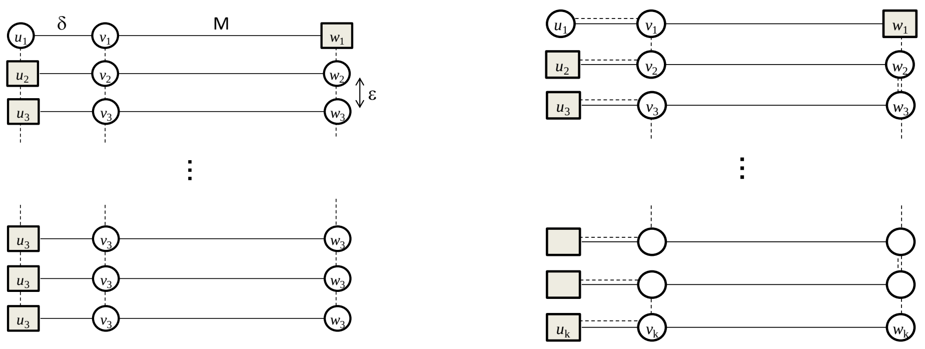
Figure 4b depicts an optimal MDRPP solution to the above instance. Small required edges (u_k, v_k) in all components different from the first one are served from their depot and are traversed twice. The *large* required edges (v_k, w_k) are traversed once in one single route associated with the depot of the first component. This route also traverses the small required edge (u_1, v_1) twice, traverses the unrequired edges (v_k, v_{k+1}) and (w_k, w_{k+1}) once if k is odd, and traverses the unrequired edges (w_k, w_{k+1}) twice if k is even. Hence, $z^*(MDRPP) = 2|K|\delta + (|K|M + 2(|K| - 1)\varepsilon)$. Therefore, for the instance described above we have

$$\frac{z^*(MDRPP_C)}{z^*(MDRPP)} = \frac{2|K|(\delta + M)}{2|K|\delta + (|K|M + 2(|K| - 1)\varepsilon)},$$

which tends to 2 when $\delta \to 0$ and $\varepsilon \to 0$. ■

3 Mathematical programming formulation and polyhedral analysis

A natural modeling option when dealing with routing problems with multiple depots is to make use of binary variables that associate arcs or edges with depots, and then define the routes of each depot. This offers two main advantages. On the one



(a) Input graph

(b) Optimal MDRPP solution

Figure 4: Potential improvement due to the modeling hypothesis H1.

hand, in absence of capacity or other type of constraints, the feasibility of a route corresponding to a fixed depot is guaranteed through the imposition of parity and connectivity constraints. On the other hand, the routes can be easily constructed once the values of the decision variables are known. The obvious disadvantage of this modeling approach is that the number of variables increases with the number of depots, and therefore the success of exact solution methods on large size instances becomes a challenge. The two MDRPP papers [14, 16] referenced in the introduction use this type of decision variables. In [16] which deals with exactly the same undirected MDRPP that we study in this paper, instances with up to 100 vertices and four depots were solved to optimality. In [14], which considers a directed MDARP dealing with carriers collaboration, instances with up to 50 vertices and two depots were optimally solved.

In this section we present a new integer linear formulation for the MDRPP with binary decision variables which are solely associated with edges, but not with depots. However, the reduction on the number of decision variables comes at the expense of additional difficulties. Figure 5, where gray squares represent depots and solid lines required edges, illustrates that connectivity and parity constraints are not sufficient to guarantee well-defined routes. Observe that the displayed solution is not feasible as it is not possible to decompose it into three routes, each starting and ending at the same depot. To overcome this difficulty we propose a new set of constraints that guarantee that each route starts and ends at the same depot, which can be separated in polynomial time.

3.1 Mathematical programming formulation

In addition, our formulation exploits the optimality conditions presented in Section 2.1, which allows the exclusive use of binary variables. In particular, two sets of binary variables are used, associated with the first and second traversals of edges, respectively. We denote by $E^y \subset E$ the set of edges that can be traversed twice in an optimal solution, which, according to O4, consists of all required edges plus the edges of T_C. For each $e \in E$, let x_e be a binary variable indicating whether or not edge e is traversed by some route. For each $e \in E^y$, let y_e be a binary variable that equal to one if and only if edge e is traversed twice. Then, an integer linear program

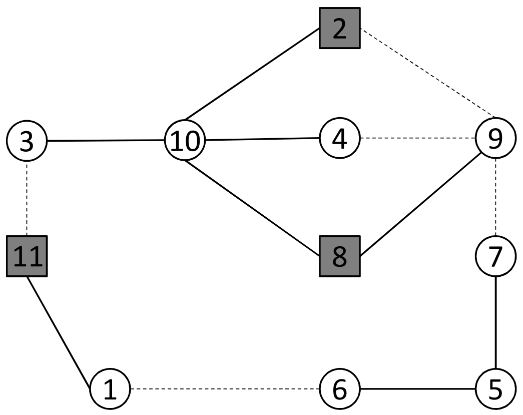


Figure 5: Infeasible solution satisfying connectivity and parity constraints

for the MDRRP is as follows:

$$\text{minimize} \sum_{e\in E} c_e x_e + \sum_{e \in E^y} c_e y_e \qquad (1)$$

subject to

$$
\begin{aligned}
&(x+y)(\delta(d)) \geq 2 && d \in D && (2)\\
&(x+y)(\delta(S)) \geq 2 && S \subseteq V \setminus\ D && (3)\\
&(x-y)(\delta(S)\setminus H) + y(H) \geq x(H) - |H| + 1 && S \subset V,\ H \subseteq \delta(S) && \\
& && |H| \text{ odd} && (4)\\
&(x-y)(Q) + y(H) \geq x(H) - |H| + |D'| && S \subset V \setminus D, D' = \{d_i\}_{i\in I} \subset D, && (5)\\
& && |D'| > 1, H_i \subseteq \delta(S) \cap \delta(d_i), |H_i| \text{ odd}, i \in I, && \\
& && H = \bigcup_{i \in I} H_i,\ Q = (\delta(S)\setminus H) \setminus \big(\delta(D\setminus D')\big) && \\
&x_e = 1 && e \in R && (6)\\
&y_e \leq x_e && e \in E^y && (7)\\
&x_e \in \{0,1\} && e \in E && (8)\\
&y_e \in \{0,1\} && e \in E^y. && (9)
\end{aligned}
$$

Constraints (2) ensure that all depots are used. Inequalities (3) are an adaptation to the case with multiple depots of the RPP connectivity inequalities (see, for instance [18]). They impose that at least two edges cross the cut-set $\delta(S)$ of any set of vertices S that does not contain a depot, i.e. $S \subset V\setminus D$. Inequalities (4) ensure the parity (even degree) of every subset of vertices and, in particular, of every vertex. They impose that if a solution uses a set H consisting of an odd number of edges incident to a set of vertices S, then the solution uses at least one additional traversal of some edge in the cut-set $\delta(S)$. In our case, we further exploit the precedence relationship of the x variables with respect to the y variables imposed by constraints (7). Therefore, the additional edge will be either a second traversal of some edge of H or a first traversal of some edge of $\delta(S)\backslash H$. Inequalities (4) are an adaptation to the MDRPP of those proposed in [2, 3, 4], which were later reinforced for the Maximum Benefit Chinese Postman Problem [10]. Inequalities (15) are new and will be referred to as *depot inequalities*. They impose that for a given subset $S \subset V\backslash D$ of vertices not including any depot, the degree of its cut-set with respect to each of the depots is even. As we will see in Proposition 3.1, this family of inequalities, jointly with the remaining connectivity and parity constraints, also guarantees that each route starts and ends at the same depot. Equalities (6) ensure that each required edge is served in the solution, whereas inequalities (7) impose that an edge cannot be traversed for a second time unless that it has been traversed for a first time. The binary conditions of the variables x and y derived from their definition are imposed by constraints (8) and (9).

The above formulation contains $|E|$ x variables and $|E^y|$ y variables. There are $|D|$ constraints of type (2), $|R|$ constraints (6), and $|E^y|$ constraints of type (7). The size of the families constraints (3), (4), and (15) is exponential in $|V|$.

Proposition 3.1 *Formulation* (2)–(9) *is valid for the MDRRP.*

Proof: We will show that if a solution $(\overline{x}, \overline{y})$ satisfying (2)–(4), (6)–(9) is not feasible for the MDRRP, then there exists a constraint (15) violated by the solution. Let $G(\overline{x}, \overline{y})$ denote the support graph associated with $(\overline{x}, \overline{y})$. Because of the connectivity and parity constraints (3)–(4), if $(\overline{x}, \overline{y})$ is not feasible for the MDRRP then there must be a simple tour T traversing at least two depots. Let $d_1, d_2 \in D$ be two consecutive depots in one of the orientations of T, and $P_{d_1 d_2}$ the subpath of T connecting d_1 and d_2. The result follows from the observation that the depot constraint (15) associated with $S = V(P_{d_1 d_2}) \setminus D$, $D' = \{d_1, d_2\}$, $H_1 = S \cap \delta(d_1)$, $H_2 = S \cap \delta(d_2)$ and $Q = (\delta(S) \setminus H) \setminus (\delta(D \setminus D'))$ is violated by $(\overline{x}, \overline{y})$. ■

Let us use again the example of Figure 5 for illustrative purposes. Consider, for instance, the simple tour $T = (11, 3, 10, 8, 9, 7, 5, 6, 1)$, $d_1 = 11$, $d_2 = 8$, and $P_{11,8} = (11, 3, 10, 8)$. Using the notation of the above proof, $V(P_{d_1 d_2}) \setminus D = \{3, 10\}$, $H_1 = \{(3, 11)\}$, and $H_2 = \{(8, 10)\}$ and $Q = (\delta(S) \setminus H) \setminus (\delta(D \setminus D')) = \{(4, 10)\}$. Indeed, the associated depot inequality (15) is violated since $\overline{x}(H) - |H| + |D'| = 2$, but $(\overline{x} - \overline{y})(Q) + \overline{y}(H) = 1 < \overline{x}(H) - |H| + |D'|$.

Remark 3.1 An additional consequence of the above proof is that depot inequalities (15) associated with subsets D' with two depots suffice to guarantee that the proposed formulation is valid.

3.2 Polyhedral analysis

Considering that all required edges must be traversed at least once, the MDRPP can be equivalently stated as the problem of determining a least cost set of additional edges which, along with the required edges, define a connected route from each depot. Accordingly, we can reformulate (2)–(9) by slightly modifying the meaning and the domain of the variables. Now the variables x_e will have a different meaning depending on whether or not e is a required edge. For $e \in R$, $x_e = 1$ indicates that required edge e is traversed one additional time (second traversal), whereas for $e \in E \setminus R$ $x_e = 1$ indicates that unrequired edge e is traversed for the first time. Based on the optimality conditions of Section 2.1, we redefine the domain of the x variables as $E^2_x \subset E$, which contains all required edges, plus the edges that satisfy condition O3, as well as the unrequired edges connecting two end-vertices in different components. Now the domain $E^2_y \subset E$ for the variables associated with the second traversal of edges, contains only the unrequired edges that satisfy condition O4, i.e. the unrequired edges of T_C. In terms of these new sets of variables, the MDRPP can be expressed as

$$\sum_{e\in R} c_e + \min\{\sum_{e\in E^2_x} c_e x_e + \sum_{e\in E^2_y} c_e y_e\} \tag{10}$$

subject to

$$
\begin{aligned}
& x(\delta(d)) \geq 2 && d \in D,\ |\delta(d)| = 1 && (11)\\
& (x+y)(\delta(S)) \geq 2 && S = \cup_{i\in K'} V_i \setminus D && \\
& && \emptyset \neq K' \subset K && (12)\\
& (x-y)(\delta(S)\setminus H) + y(H) \geq x(H) - |H| + 1 && S \subset V,\ S\ R\text{-even} && \\
& && H \subseteq \delta(S),\ |H| \text{ odd} && (13)\\
& (x-y)(\delta(S)) \geq 1 && S \subset V,\ S\ R\text{-odd} && (14)\\
& (x-y)(Q) + y(H) \geq x(H) - |H| + |D'| && S \subset V\setminus D, D' = \{d_i\}_{i\in I} \subset D, && (15)\\
& && |D'| > 1, H_i \subseteq \delta(S)\cap\delta(d_i), |H_i| \text{ odd}, i \in I, && \\
& && H = \bigcup_{i\in I} H_i,\ Q = (\delta(S)\setminus H)\setminus\big(\delta(D\setminus D')\big) && \\
& y_e \leq x_e && e \in E^2_y && (16)\\
& x_e \in \{0,1\} && e \in E^2_x && (17)\\
& y_e \in \{0,1\} && e \in E^2_y. && (18)
\end{aligned}
$$

Next we study the polyhedral properties of (10)-(18). We denote by $P_{(MDRPP)}$ the polytope defined by the convex hull of feasible solutions to the above formulation:

$P_{(MDRPP)} = conv\{(x,y) \in \{0,1\}^{|E^2_x|+|E^2_y|} : (x,y) \text{ satisfies (11)–(16)}\}$.

In the proofs below we abuse notation and assume that there exists an edge connecting each pair of vertices. When such edges are *non-existing* in E^2_x, they correspond to T-joins, connecting given pairs of vertices, that only use *true* edges of the set E^2_x. Examples of such non-existing edges are, for instance, T-joins connecting two depots, or T-joins connecting two even-vertices in the same component if the connecting edges do not exist in E^2_x. Using edges associated with such T-joins in the solutions that we will build, will simplify the presentation of the proofs, but will have no effect on their validity, since the parity of the intermediate vertices in the T-joins will not be affected.

Proposition 3.2 *$P_{(MDRPP)}$ is full-dimensional if and only if every cut-edge set $\delta(S) \subset V \setminus D$ has at least three edges, and every cut-edge set $\delta(S)$ such that $S = \bigcup_{i\in K'} V_i \setminus D$ ($\emptyset \neq K' \subset K$) has at least four edges, where if $e \in E_x^2$ and $e \in E_y^2$, then e is counted as two distinct edges.*

Proof: *The condition is necessary.* We follow the same idea as in [18] for the RPP. If there exists a cut edge-set with only one edge, then e should be a required edge and $x_e = 1$. Therefore, $P(MDRPP) \subset \{x : x_e = 1\}$. Assume now there exists a subset $S \subset V \setminus D$, with $\delta(S) = \{e^{(1)}, e^{(2)}\}$. If $S = \cup_{i\in K'} V_i \setminus D$ ($\emptyset \neq K' \subset K$), then $P(MDRPP) \subset \{x : x_{e^{(1)}} = 1 \text{ and } x_{e^{(2)}} = 1\}$. Otherwise, if $\delta(S)$ is R-even, $P(MDRPP) \subset \{x : x_{e^{(1)}} = x_{e^{(2)}}\}$, and if $\delta(S)$ is R-odd, $P(MDRPP) \subset \{x : x_{e^{(1)}} + x_{e^{(2)}} = 1\}$. Finally, if $S = \cup_{i\in K'} V_i \setminus D$ ($\emptyset \neq K' \subset K$) or $\delta(S) = \{e^{(1)}, e^{(2)}, e^{(3)}\}$, then $P(MDRPP) \subset \{x : x_{e^{(1)}} + x_{e^{(2)}} + x_{e^{(3)}} = 2\}$

The condition is sufficient. Let us find $|E_x^2| + |E_y^2| + 1$ affinely independent solutions satisfying the connectivity, parity and depot inequalities.

The first solution, denoted by (x^0, y^0), contains one traversal of the edge connecting each R-odd vertex with an arbitrarily chosen depot $d_0 \in D$, plus two traversals of all the edges of T_C (O4). To guarantee the parity of the depots in the solution, it may be necessary to add some edges connecting some pairs of depots. The remaining $|E_x^2| + |E_y^2|$ solutions, (x^e, y^e), $e \in E_x^2 \cup E_y^2$, are obtained from (x^0, y^0) as follows:

a) Case $e = (u, v) \in E_x^2$, with u, v in the same component.

- *a*1) Case u, v R-odd. In this case $x_e^0 = 0$ whereas the components corresponding to edges $e_u = (d_0, u)$ and $e_v = (d_0, v)$, take the value 1, i.e. $x_{e_u}^0 = x_{e_v}^0 = 1$. We set $x_e^e = 1 - x_e^0 = 1$, $x_{e_u}^e = 1 - x_{e_u}^0 = 0$, and $x_{e_v}^e = 1 - x_{e_v}^0 = 0$, so the parity of u and v does not change. All other components remain unchanged, i.e., $x_f^e = x_f^0$, for all $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y$.
- *a*2) Case u, v R-even. In this case $x_e^0 = 0$ whereas the components corresponding to edges $e_u = (d_0, u)$ and $e_v = (d_0, v)$, take the value 0, i.e. $x_{e_u}^0 = x_{e_v}^0 = 0$. We set $x_e^e = 1 - x_e^0 = 1$, $x_{e_u}^e = 1 - x_{e_u}^0 = 1$, and $x_{e_v}^e = 1 - x_{e_v}^0 = 1$, so the parity of u and v does not change. All other components remain unchanged, i.e., $x_f^e = x_f^0$, for all $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y$.
- *a*3) Case u R-odd and v R-even (or vice versa). In this case $x_e^0 = 0$ whereas the components corresponding to edges $e_u = (d_0, u)$ take the value 1 and $e_v = (d_0, v)$, take the value 0, i.e. $x_{e_u}^0 = 1$ and $x_{e_v}^0 = 0$. We set $x_e^e = 1 - x_e^0 = 1$, $x_{e_u}^e = 1 - x_{e_u}^0 = 0$, and $x_{e_v}^e = 1 - x_{e_v}^0 = 1$, so the parity of u and v does not change. All other components remain unchanged, i.e., $x_f^e = x_f^0$, for all $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y$.

b) Case $e = (u, v) \in E_x^2 \setminus E_y^2$, with u, v in different components. In this case $x_e^0 = 0$, and the components corresponding to edges $e_u = (d_0, u)$ and $e_v = (d_0, v)$, are at value 0 as well, i.e. $x_{e_u}^0 = x_{e_v}^0 = 0$. Again we set $x_e^e = 1 - x_e^0$, $x_{e_u}^e = 1 - x_{e_u}^0$, and $x_{e_v}^e = 1 - x_{e_v}^0$, resulting now in $x_e^e = x_{e_u}^e = x_{e_v}^e = 1$. As in the previous case, the parity of u and v does not change. All other components remain unchanged, i.e., $x^e_f = x^0_f$, for all $f \in E^2_x \setminus \{e\}$, $y^e_f = y^0_f$, for all $f \in E^2_y$.

c) Case $e \in E^2_x \cap E^2_y$. Now $x^0_e = y^0_e = 1$. We now generate two solutions: (x^e, y^e), associated with $e \in E^2_x$, and (x'^e, y'^e), associated with $e \in E^2_y$. For (x^e, y^e) we keep $x^e_e = 1$ but set $y^e_e = 0$. Then we set $x^e_{e_u} = x^e_{e_v} = 1$ where, as before, u, v denote the two end-vertices of edge e, and $e_u = (d_0, u)$, $e_v = (d_0, v)$. This guarantees the parity of u and v and the connectivity of the solution. All other components remain unchanged. For (x'^e, y'^e) we set $x'^e_e = y'^e_e = 0$. This guarantees the parity of u and v although the connectivity may be lost. If needed, additional edges are added to recover connectivity (indeed it is possible to recover connectivity via a triangle of edges connecting u and v to an arbitrary depot). All other components remain unchanged.

Note that each of the feasible solutions obtained above contains at least one component with a value that is different from the values of that component in all other solutions. Therefore, the associated points are affinely independent and the result follows. ■

In the remainder of this section we study the conditions under which several families of inequalities define facets of $P_{(MDRPP)}$. The proofs of these results are similar or adaptations of those for the RPP in [18].

Proposition 3.3 *The inequality $x_e \geq 0$ defines a facet of $P_{(MDRPP)}$ if and only if every cut-set $\delta(S) \subset V \setminus D$ containing e has at least four edges and every $\delta(S)$ such that $S = \bigcup_{i \in K'} V_i \setminus D$ ($\emptyset \neq K' \subset K$) has at least five edges.*

Proof: The proof in [18] directly applies to the MDRPP, independently of the number of depots. The face $\{x \in P_{(MDRPP)} : x_e = 0\}$ has the same dimension as the polytope associated with the MDRPP defined on the graph obtained after removing edge e from G. ■

Proposition 3.4 *The inequality $x_e \leq 1$ induces a facet of $P_{(MDRPP)}$ if and only if every cut-set $\delta(S)$ containing e has at least four edges.*

Proof: *The condition is necessary.* Suppose there exists a cut-edge set with only three edges, $\delta(S) = \{e, f, g\}$. Then, either $\{x \in P_{(MDRPP)} : x_e = 1\} \subset \{x \in P_{(MDRPP)} : x_e = 1, x_f + x_g = 1\}$ if $\delta(S)$ is R-even, or $\{x \in P_{(MDRPP)} : x_e = 1\} \subset \{x \in P_{(MDRPP)} : x_e = 1, x_f - x_g = 0\}$ otherwise.

The condition is sufficient. Under the hypotheses, it is easy to show that there exist $|E^2_x| + |E^2_y|$ feasible and affinely independent solutions on the hyperplane $x_e = 1$.

Let the first solution be solution (x^e, y^e) as defined in the proof of Proposition 3.2. Recall that $x^e_e = 1$. The remaining $|E^2_x| + |E^2_y| - 1$ solutions may be constructed following the same process as in Proposition 3.2, modifying in each new solution one of the other components. ■

Proposition 3.5 *The connectivity inequality* (12) *associated with $S = \bigcup_{i \in K'} V_i$ ($\emptyset \neq K' \subset K$), $S \bigcap D = \emptyset$, induces a facet of $P_{(MDRPP)}$ if and only if the graphs* *induced by the connected components $G(S)$ and $G(V\backslash S)$ verify the following: i) $G(S)$ is connected and each connected component of $G(V\backslash S)$ has at least one depot. ii) For every subset of components in $S' \subset S$ (or S' in $V\backslash S$) with $S' \bigcap D = \emptyset$, it holds that $|\delta(S')\backslash\delta(S)| \geq 2$.*

Proof: *The condition is necessary.* Suppose $G(S)$ is not connected, and let S_1 be a component of $G(S)$. Then the connectivity inequality (12) associated with $G(S)$ is dominated by the connectivity inequality (12) corresponding to $G(S_1)$. A similar situation arises if some component of $G(V\backslash S)$ contains no depot. Suppose now there exists a subset of components $S' \subset S$ such that there is only one edge connecting S' and $S\backslash S'$. Then, the connectivity constraint associated with $G(S)$ is dominated by the sum of the connectivity constraints (12) associated with S' and $S\backslash S'$.

The condition is sufficient. Under the hypotheses, there exist $|E_x^2| + |E_y^2|$ feasible and affinely independent solutions on the hyperplane $\sum_{e\in\delta(S)}(x_e + y_e) = 2$.

Consider a solution (x^0, y^0) that contains: *i)* one traversal of the edges connecting each R-odd vertex with an arbitrarily even vertex $i \in V$ in its own component; *ii)* two traversals of one arbitrarily selected edge of T_C, $e_0 = (u_0, v_0)$, which belongs to the cut-set $\delta(S)$, i.e. $e_0 \in E_y^2 \cap \delta(S)$; and, *iii)* two traversals of all the edges of T_C that do not belong to the cut-set $\delta(S)$. By construction, $(x^0 + y^0)(\delta(S)) = 2$. The $|E_x^2| + |E_y^2| - 1$ additional solutions are obtained from (x^0, y^0) as follows:

a) Case $e = (u, v) \in E_x^2$, with u, v in the same component. We proceed exactly as in case *a)* in the proof of Proposition 3.2.

  *a*1) Case u, v R-odd. In this case $x_e^0 = 0$ whereas the components corresponding to edges $e_u = (i, u)$ and $e_v = (i, v)$, take the value 1, i.e. $x_{e_u}^0 = x_{e_v}^0 = 1$. Hence, we set $x_e^e = 1$ and $x_{e_u}^e = x_{e_v}^e = 0$, so the parity of u and v does not change. All other components remain unchanged, i.e., $x_f^e = x_f^0$, for all $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y^2$.

  *a*2) Case u, v R-even. In this case $x_e^0 = 0$ whereas the components corresponding to edges $e_u = (i, u)$ and $e_v = (i, v)$, take the value 0, i.e. $x_{e_u}^0 = x_{e_v}^0 = 0$. We set $x_e^e = 1 - x_e^0 = 1$, $x_{e_u}^e = 1 - x_{e_u}^0 = 1$, and $x_{e_v}^e = 1 - x_{e_v}^0 = 1$, so the parity of u and v does not change. All other components remain unchanged, i.e., $x_f^e = x_f^0$, for all $f \in E_x^2\backslash\{e\}$, $y_f^e = y_f^0$, for all $f \in E_y$.

  *a*3) Case u R-odd and v R-even (or vice versa). In this case $x_e^0 = 0$ whereas the components corresponding to edges $e_u = (i, u)$ take the value 1 and to edges $e_v = (i, v)$ take the value 0, i.e. $x_{e_u}^0 = 1$ and $x_{e_v}^0 = 0$. We set $x_e^e = 1 - x_e^0 = 1$, $x_{e_u}^e = 1 - x_{e_u}^0 = 0$, and $x_{e_v}^e = 1 - x_{e_v}^0 = 1$, so the parity of u and v does not change. All other components remain unchanged, i.e., $x_f^e = x_f^0$, for all $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y$.

b) Case $e = (u, v) \in E_x^2 \setminus E_y^2$ with u, v in different components. In this case $x_e^0 = 0$, and there is an edge $e' = (u', v') \in E_y \cap \delta(S)$ with u' in the same component as u, and v' in the same component as v with $x_{e'}^0 = y_{e'}^0 = 1$. Note that it is possible that u' coincides with u or that v' coincides with v (but not both simultaneously). Now we set $x^e_e = 1$, $y^e_{e'} = 0$. Furthermore, if $u' \neq u$ we set $x^e_{e_{u,u'}} = 1$. Similarly, $x^e_{e_{v,v'}} = 1$, provided that $v' \neq v$. Like in the previous case, the parity of u and v does not change. All other components remain unchanged (including $x^e_{e'} = 1$).

c) Case $e = (u, v) \in E^2_y \setminus \{e_0\}$. Now $x^0_e = y^0_e = 1$. We now generate two solutions: (x^e, y^e), associated with $e \in E^2_x$, and (x'^e, y'^e), associated with $e \in E^2_y$. Consider the following subcases:

c1) Case $e = (u, v) \in \delta(S)$. For (x^e, y^e) we set $x^e_{e_0} = y^e_{e_0} = 0$ and $x^e_e = y^e_e = 1$. For (x'^e, y'^e), we set $x'^e_{e_0} = y'^e_{e_0} = 0$, $x'^e_e = 1$, and $y'^e = 0$. To recover the parity at the end-vertices of e and to guarantee the connectivity of the new solution and $(x'^e + y'^e)(\delta(S)) = 2$ we use edges $e^u = (i, u)$ and $e^v = (i, v)$, and set $x'^e_{e^u} = x'^e_{e^v} = 1$. All other components remain unchanged.

c2) Case $e = (u, v) \notin \delta(S)$. For (x^e, y^e), we keep $x^e_{e_0} = y^e_{e_0} = 1$ and set $x^e_e = y^e_e = 0$. Without loss of generality, we assume that u is the end-vertex in the part that would be disconnected from the part of the solution containing e_0, if all other components remained unchanged. This solution guarantees the parity of the vertices, but the connectivity may be lost if some of the two split parts contains no depot. In this case, to recover the connectivity of the solution we use edges $e^{u_0} = (u, u_0)$ and $e^{v_0} = (u, v_0)$, and set $x^e_{e^{u_0}} = x^e_{e^{v_0}} = 1$. All other components remain unchanged. For (x'^e, y'^e), again we keep $x'^e_{e_0} = y'^e_{e_0} = 1$, but we now set $x'^e_e = 1$ and $y'^e_e = 0$. To recover the parity of u and v, we now use the edges that connect each of them with some vertex i, denoted by $e^u = (u, i)$ and $e^v = (v, i)$, and set $x'^e_{e^u} = y'^e_{e^v} = 1$. All other components remain unchanged. ■

Proposition 3.6 *The parity inequalities* (13) *and* (14) *induce facets of* $P_{(MDRPP)}$ *if and only if the following conditions hold: i) for every subset* $S' \subset S$ *(or* S' *in* $V \backslash S$*) with* $S' \cap D = \emptyset$*, then* $|\delta(S') \backslash \delta(S)| \geq 2$*. ii) If* $|H| = 1$*, then* S *is not a set of components (*S *cannot be expressed as* $S = \cup_{i \in K'} V_i \setminus D$ *with* $\emptyset \neq K' \subset K$*).*

Proof: The proof in [18], based on [5], directly applies to the MDRPP, independently of the number of depots. ■

Proposition 3.7 *The depot inequalities* (**??**) *induce facets of* $P_{(MDRPP)}$ *if and only if for every subset* $S' \subset S$ *(or* $S' \subset V \backslash S$*) with* $S' \cap D = \emptyset$*, then* $|\delta(S') \backslash \delta(S)| \geq 2$*.*

Proof: The depot inequalities are an adaptation of the parity inequalities. So, as in the previous case, the proof in [18], based on [5], directly applies to the depot inequalities. ■

Remark that in Proposition 3.7 condition *ii)* of Proposition 3.6 is no longer needed. The reason is that inequalities (**??**) are defined for $r > 1$. Therefore, they are never not dominated by the connectivity inequality (12) associated with S, as it happens for inequalities (13) when condition *ii)* does not hold.

4 Branch-and-cut algorithm

In this section, we present an exact branch-and-cut algorithm for the MDRPP, based on formulation (2)–(9) proposed in Section 3. As usual, the families of exponential size, connectivity, parity and depot constraints, (3), (4) and (15), respectively, are initially relaxed. Then, after each LP iteration they are separated to detect whether there are constraints of any of these families violated by the current LP solution. If so, the detected violated constraints are incorporated to the current formulation, and the reinforced formulation is solved.

We use the Mixed Integer Linear Programming solver of CPLEX for setting the enumeration framework. Thus, the above strategy is embedded within an enumeration scheme, which means that it is applied not only at the root node of the enumeration tree but also at all generated nodes. We now describe the main elements of the algorithm: the initial relaxation and the separation procedures.

4.1 Initial relaxation

The algorithm starts with all integrality conditions relaxed and only a subset of constraints. In the initial formulation we include all constraints (2), (6) and (7), plus a small subset of connectivity and parity constraints. In particular, we consider two small subfamilies of the connectivity constraints (3): on the one hand, the inequalities associated with the subsets defined by the end-vertices of the edges not incident with any depot, i.e., $S = \{u, v \in V \mid (u, v) \in E \text{ and } u, v \in V \setminus D\}$; on the other hand, the inequalities associated with the subsets defined by the vertices of each component without depot, i.e. $S = V_k$, $k \in K$, with $V_k \cap D = \emptyset$. As for the parity constraints (4), initially, we only consider the ones associated with R-odd singletons i.e., $S = \{v\}$ with $v \in V$ and $|\delta_R(v)|$ odd.

4.2 Separation of inequalities

Let $G(\overline{x}, \overline{y})$ denote the support graph associated with the LP solution $(\overline{x}, \overline{y})$ at any iteration of the algorithm. Inequalities (15) are only separated when the LP solution $(\overline{x}, \overline{y})$ is integer, while inequalities (3) and (4) are also separated when the LP solution $(\overline{x}, \overline{y})$ is fractional. In each case we proceed as follows.

a) *Case* $(\overline{x}, \overline{y})$ *is integer:* Check for violated inequalities of types (3), (4) and (15).

- $a1$) Connectivity inequalities (3). Violated inequalities can be identified by finding connected components of $G(\overline{x}, \overline{y})$ containing no depot. The vertex set of each such component defines a violated cut.
- $a2$) Parity inequalities (4). Violated inequalities can be identified by checking the parity of each vertex. In this case each vertex $v \in V$ with $|(\overline{x} + \overline{y})(\delta(E_{(v)})|$ odd defines a violated cut.
- $a3$) Depot inequalities (15). Violated inequalities can be easily identified by first finding a tour decomposition of the solution (applying, for instance,

Hierholzer's algorithm [22]) and then checking if any of the tours contains a path $P_{d_1d_2}$ connecting two (consecutive) depots. In this case $D' = \{d_1, d_2\}$ and $S = V(P_{d_1d_2}) \setminus D'$ defines a violated cut.

b) Case $(\overline{x}, \overline{y})$ *is fractional:* Check for violated inequalities of types (3) and (4).

In each case, we first apply a heuristic and only resort to exact separation when the heuristic fails. For parity inequalities exact separation is only applied if, in addition, some parity cut has been added in the last ten iterations and the value of the objective function has increased by at least φ from the previous iteration, where φ is a given parameter. For both types of inequalities, the heuristic looks for connected components in an ad hoc graph. Heuristic and exact separation for each case are described below.

4.2.1 Separation of connectivity inequalities (3)

The separation for inequalities (3) is to find $S \subset V \setminus D$, with $(\overline{x} + \overline{y})(\delta(S)) < 2$, or to prove that no such inequality exists. As the example of Figure 6 shows, violated connectivity constraints (3) are not necessarily associated with minimum cuts in $G(\overline{x}, \overline{y})$ relative to the capacities vector $\overline{x} + \overline{y}$. Thus, for solving the separation problem for constraints (3) we cannot apply the usual technique, consisting of identifying the tree of minimum cuts for $G(\overline{x}, \overline{y})$ relative to $\overline{x} + \overline{y}$. Instead, we will operate on the subgraph $G^{V\setminus D}(\overline{x}, \overline{y})$ induced by the vertex set $V \setminus D$ and look for minimum cutsets relative to $\overline{x} + \overline{y}$. Indeed the value of such cut-sets for $G^{V\setminus D}(\overline{x}, \overline{y})$ need not correspond to their real value in $G(\overline{x}, \overline{y})$. Nevertheless if $v^{V\setminus D}(S)$ denotes the value the min-cut of a vertex set $S \subset V \setminus D$ for $G^{V\setminus D}(\overline{x}, \overline{y})$, then the real value for $G(\overline{x}, \overline{y})$ can be easily computed as $(\overline{x} + \overline{y})(\delta(S)) = v^{V\setminus D}(S) + (\overline{x} + \overline{y})\,(\delta(S) \cap \delta(D))$.

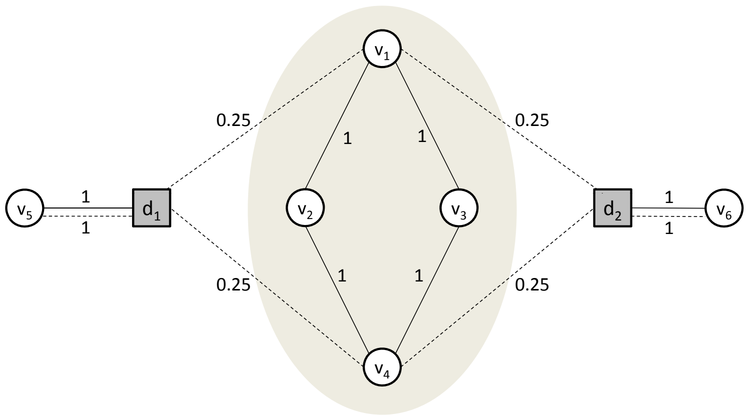


Figure 6: Violated connectivity (3) not associated with a minimum cut in $G(\overline{x}, \overline{y})$

Heuristic separation

The heuristic for the separation of (3) looks for connected components in the subgraph of $G^{V\setminus D}(\overline{x}, \overline{y})$, that contains only those edges with values $\overline{x_e} + \overline{y_e} \geq \varepsilon$, where ε is a given parameter. Then, we compute the *real* value of the cut associated with each connected component C, which, by construction, contains no depots. If $(\overline{x}+\overline{y})\,(\delta(V(C)))<2$, the connectivity inequality (3) associated with $V(C)$ is violated by $(\overline{x},\overline{y})$.

Exact separation
For the exact separation of connectivity constraints (3) we build the tree of min-cuts T of $G^{V\setminus D}(\overline{x},\overline{y})$ (see, for instance, [20]) with capacities given by $\overline{x}_e+\overline{y}_e$. Since $(\overline{x}+\overline{y})(\delta(S))=v^{V\setminus D}(S)+(\overline{x}+\overline{y})\,(\delta(S)\cap\delta(D))$, the minimum cut-set of T is not necessarily the cut-set with a minimum value of $(\overline{x}+\overline{y})(\delta(S))$. Thus, it may be necessary to check more than one min-cut of T. In particular, for each min-cut $\delta(S)$ of T of value $v^{V\setminus D}(S)<2$, we check if $v^{V\setminus D}(S)+(\overline{x}+\overline{y})\,(\delta(S)\cap\delta(D))<2$ as well. In this case the inequality (3) associated with S is violated by $(\overline{x},\overline{y})$.

4.2.2 Separation of parity inequalities (4)

The separation problem for inequalities (4) is to find $S\subset V$, $H\subseteq\delta(S)$, $|H|$ odd such that

$$(\overline{x}-\overline{y})(\delta(S)\setminus H)+\overline{y}(H)<\overline{x}(H)-|H|+1, \tag{19}$$

or to prove that no such inequality exists. The procedure we use follows the spirit of the separation used by other authors with similar parity constraints for other arc routing problems with binary variables [2, 3, 8]. Note that (19) can be written as

$$(\overline{x}-\overline{y})(\delta(S)\setminus H)+|H|-(\overline{x}-\overline{y})(H)<1. \tag{20}$$

or equivalently as

$$\sum_{e\in\delta(S)\setminus H}(\overline{x}_e-\overline{y}_e)+\sum_{e\in H}(1-(\overline{x}_e-\overline{y}_e))<1. \tag{21}$$

The above expression indicates that for a given set $S\subset V$, a set $H\subset\delta(S)$ that yields the smallest value in the left-hand side is given by $H=\{e\in\delta(S)\mid 1-(\overline{x}_e-\overline{y}_e)<\overline{x}_e-\overline{y}_e\}$. Further, the value of the left-hand side of (20) corresponds to the value of the cut-set $\delta(S)$ relative to a capacities vector $b_e=\min\{(\overline{x}_e-\overline{y}_e),1-(\overline{x}_e-\overline{y}_e)\}$. Hence, the vertex set $S\subset V$, and associated edge set $H\subset\delta(S)$, which minimize the left-hand side of (20) can be obtained by finding the minimum cut in $G_{\overline{x},\overline{y}}$ relative to the capacities vector b_e as defined above. Indeed, for a given set S the set $H\subset\delta(S)$ defined above need not be odd. If $|H|$ is even, the smallest increment in the value of the left-hand side of (20) that guarantees that $|H|$ is odd is obtained by either removing one edge from $|H|$ (and transferring it to $\delta(S)\setminus H$) or by adding to H one edge currently in $\delta(S)\setminus H$. In particular, the smallest increment is obtained with

$$\Delta=\min\left\{\min\{\overline{x}_e-\overline{y}_e:e\in\delta(S)\setminus H\},\min\{1-(\overline{x}_e-\overline{y}_e):e\in H\}\right\}.$$

When $b(\delta(S))+\Delta<1$, the updated set $|H|$ defines a violated inequality (4) for S in the current solution $(\overline{x},\overline{y})$.

Heuristic separation
The heuristic consists of finding the connected components in the subgraph $G(\overline{x},\overline{y})$

induced by edges with values $b_e = \min\{(\overline{x}_e - \overline{y}_e), 1 - (\overline{x}_e - \overline{y}_e)\} > \varepsilon$, where ε is a given parameter. Then, if $S \subset V$ is the vertex set of one of the components, we proceed as indicated above to identify its associated edge set H. If $b(\delta(S)) < 1$ and $|H|$ is odd, then the parity constraint (4) associated with S and H is violated by $(\overline{x}, \overline{y})$. Otherwise, if $b(\delta(S)) + \Delta < 1$, the parity constraint (4) associated with S and the updated set H is violated by $(\overline{x}, \overline{y})$. If $|H|$ is odd and $b(\delta(S)) + \Delta \geq 1$, then the heuristic fails.

Exact separation
For the exact separation of the parity constraints (4) we build the tree of min-cuts T^b of $G_{\overline{x},\overline{y}}$ with capacities given by b_e. Let $S^1, ..., S^r$ be the vertex sets of the minimum cuts of T^b with values $v^{S^i} = b(\delta(S^i))$ strictly smaller than one, ordered by non-decreasing order of their values, i.e., $v^{S^1} \leq \cdots \leq v^{S^r} < 1$. Then we proceed as follows:

```
end ← false; i ← 1
while $v^{S^i} < 1$ and end = false do
    Define $H^i \subset \delta(S^i)$ as explained above.
    if($|H^i|$ odd then )
        end ← true (constraint (4) violated by $(\overline{x}, \overline{y})$ for $S^i$ and $H^i$ )
    else
        Compute $\Delta = \min\left\{\min\{\overline{x}_e - \overline{y}_e : e \in \delta(S^i) \setminus H^i\}, \min\{1 - (\overline{x}_e - \overline{y}_e) : e \in H^i\}\right\}$
        if ($v^{S^i} + \Delta < 1$) then
            end ← true
            (constraint (4) violated by $(\overline{x}, \overline{y})$ for $S^i$ and updated set $H^i$)
        else
            if ($i = r$) then
                end ← true        (no violated constraint (4) exists)
            else
                $i \leftarrow i + 1$
            end-if
        end-if
    end-if
```

Summarizing, the exact separation for inequalities (4) reduces to finding the set S such that $\delta(S)$ contains the best possible set H, and indicates that in the worst case, this problem can be solved by finding the the complete tree of min-cuts of the support graph $G(\overline{x}, \overline{y})$, for the capacities vector b defined above. It is important to recall that the smallest value of the left-hand side of inequality (4) after making H odd is not necessarily associated with the smallest min-cut of the tree.

5 Computational Experiments

The branch-and-cut algorithm was coded in C++ using CPLEX 12.5 Concert Technology for the solution of the LP relaxations. Default parameters were used, except for the maximum computing time, which has been set to 24 hours per instance, and the cuts generated by CPLEX, which have been disabled. After some tuning, we set the value $\varphi = 0.25$ for the parameter that indicates whether or not to apply exact separation for the parity inequalities (4). The values for the threshold ε used in the heuristic separation of constraints (3) and (4) are $\varepsilon \in \{0.1, 0.2\}$ and $\varepsilon \in \{0.1, 0.2, 0.3\}$, respectively. In both cases the heuristic starts with the largest value and decreases it to the next value if it fails. The code was run on an Intel Core 2 CPU, 2.67 GHz and 8.00 GB RAM.

The algorithm was tested on two sets of benchmark instances. Both were adapted to the MDRPP from well-known sets of RPP benchmark instances. The first set was already used in [16] and contains 118 instances with up to 100 vertices. These instances were adapted from the AlbaidaA and AlbaidaB instances [12, 13]; the 24 "P" instances of [7]; the 36 instances with vertices of degree 4 and disconnected required edges (labeled "D"), 36 grid instances (labeled "G"), and the 20 randomly generated instances (labeled "R") [21]. The new set of benchmark instances contains larger instances with up to 750 vertices, which have been adapted from the "ALBA", "GRP" and "MADR" GRP instances and from the "URP" URPP instances [11] available at http://www.uv.es/corberan/instancias.htm.

In all cases we inherited the set of required edges and the cost function c from the original instances. We have considered two different cases: two and four depots. The depots were chosen randomly from the set of vertices, ensuring that no connected component has more than one depot. For this, for each selected number of depots $|D| \in \{2, 4\}$ we have proceeded as follows. First, we randomly generated $|D|$ different numbers k_i, $i = 1, \ldots, |D|$, from a discrete uniform distribution $U[1, |K|]$, which gives the indices of the clusters were the depots are located. Then, for each selected cluster, k_i the index of the node of V_{k_i} that becomes the depot was obtained by randomly generating a number v_i from an integer uniform distribution $U[1, |V_{k_i}|]$. In order to compare the results obtained with two and four depots, the instances with fewer than four connected components were removed. In total we have 103 instances of small and medium size in the first group and 52 larger instances in the second group. For each instance there are two variants: one with two depots and another with four depots. The adapted instances are available at http://www.eio.upc.edu/en/homepages/elena/mdarp-instances.

Table 1 provides information on the instances, which are grouped according to their characteristics and sizes. The column headings are as follows: # *inst* gives the number of instances in the group after and before eliminating the instances with fewer than four components; $|V_0|$ and $|E_0|$ give, respectively, the number of vertices and edges of the original graph; the columns under $|R|$ and $|K|$ give, respectively, the number of required edges and the number of connected components in the graph induced by the required edges. When not all the instances of the group have the same value, the minimum and maximum values of the group are given. The remaining columns in the table give information on the effect of the graph transformation. In particular, $|V|/|V_0|$ and $|E|/|E_0|$ correspond to the average ratio between the number of vertices or edges, respectively, in the transformed graph relative to the number of vertices or edges in the original graph. As is often the case, the transformed graph is considerably smaller than the original graph, in terms of the number of vertices

Table 1: Summary of the instances

	# inst	$\|V_0\|$	$\|E_0\|$	$\|R\|$	$\|K\|$	$\|V\|/\|V_0\|$	$\|E\|/\|E_0\|$
ALB	2/2	90–102	144–160	88–99	10–11	1.00	30.0
P	17/24	7–50	13–184	7–78	2–8	1.00	6.55
D16	6/9	16	31	3–16	2–5	0.72	3.87
D36	9/9	36	72	10–38	4–11	0.78	8.75
D64	9/9	64	128	27–75	5–15	0.82	15.75
D100	9/9	100	200	50–121	9–22	0.85	24.75
G16	7/9	16	24	3–13	3–5	0.68	5.00
G36	9/9	36	60	11–35	5–9	0.79	10.50
G64	9/9	64	112	24–68	4–14	0.80	18.00
G100	9/9	100	180	41–113	4–20	0.83	27.50
R20	2/5	20	47–75	3–7	3–4	0.41	3.29
R30	5/5	30	70-112	7–11	4–6	0.47	4.83
R40	5/5	40	82–203	8–18	5–9	0.50	6.15
R50	5/5	50	130–203	13–20	6–12	0.50	7.02
ALB2	14/15	116	174	44–119	4–23	0.78	0.85
GRP	10/10	116	174	52–126	4–34	0.76	0.83
MAD	12/15	196	316	95–219	6–42	0.81	0.91
URP5	8/12	298–493	597–1403	206–672	5–99	1.00	1.00
URP7	8/12	452–744	915–2089	321–1003	16–140	1.00	1.00

and edges. As will be explained in Section 5.1 a postprocess has been applied to the new set of larger instances after completing the input graph, as described at the beginning of Section 2, in order to reduce the number of edges of the instances. This explains the smaller values of the ratios $|E|/|E_0|$ for the new set of instances.

The results for the instances with two and four depots used in [16] are summarized in Tables 2 and 3, respectively. For each group of instances, columns 2–6 give information about the root node of the enumeration tree, while columns 7–11 give the results of the search tree. The column under $\#opt_0$ shows the number of instances in the group that were optimally solved at the root node. The column under gap_0 gives the average percentage gap at the root node with respect to the optimal or best known solution at termination. The following three columns, under the headings *cutsC*, *cutsP*, and *cutsD*, give the average number of connectivity (3), parity (4), and depot (15) cuts generated, respectively. Similarly, the next five columns under *#opt*, *gap*, *cutsC*, *cutsP*, and *cutsD* give the same information at termination: the number of instances solved to optimality, the average percent gap with respect to the optimal or best-known solution and the average number of connectivity, parity and depot cuts generated, respectively. Column under $\#Nod$ shows the average number of nodes explored in the search tree. Finally, the column under

Table 2: Computational results for instances with two depots

	$\#opt_0$	gap_0	$cutC_0$	$cutP_0$	$cutD_0$	$\#opt$	gap	cutC	cutP	cutD	$\#Nod$	CPU
ALB	1/2	0.64	12	120	0	2/2	0	3	93	0	6	45.67
P	8/17	2.35	5.82	21.24	0.35	17/17	0	1.00	36.00	0 .18	1.81	1.01
D16	3/6	0.60	2.83	8.67	0.50	6/6	0	0	1.17	0.17	1.33	0.03
D36	7/9	4.82	6.44	27.89	0.33	9/9	0	0	2.11	0	0.56	0.22
D64	6/9	2.65	7.67	36.11	0	9/9	0	1.00	13.44	0.11	1.44	1.36
D100	2/9	0.79	15.33	89.33	0	9/9	0	7.00	219.89	0.22	15.56	104.60
G16	6/7	0.71	2.14	6.57	1.14	7/7	0	0	0.43	0.14	1.29	0.02
G36	6/9	0.72	6.00	17.11	1.33	9/9	0	1.00	5.56	0.44	1.67	0.17
G64	7/9	0.88	7.44	22.56	0.11	9/9	0	1.00	6.44	0.67	1.78	0.52
G100	0/9	1.48	11.78	37.00	0.67	9/9	0	4.00	37.00	0.22	9.00	6.67
R20	2/2	0	1.00	1.50	0.50	-	-	-	-	-	0	0.01
R30	3/5	8.72	2.40	9.60	0.80	5/5	0	0	1.80	0.4	1.20	0.02
R40	5/5	0	6.00	12.20	0	-	-	-	-	-	0	0.06
R50	4/5	1.88	4.80	10.80	0	5/5	0	0	0	0.20	0.20	0.06

CPU gives the total computing time in seconds.

At can be seen, the optimality of the current solution was proven for all the instances, independently of the number of depots. Optimality was proven at the root node for, respectively, 60 two-depot and 62 four-depot instances out of the 103 instances considered in each case.

The computational effort required for solving the instances to optimality can be evaluated by the required computing times. In this sense, the optimal solution for nearly all the instances in this set, both with two and four depots, was found within less than one minute. Among the two-depot instances, the exception were one ALB, which required 70 seconds, and two D100 instances, which required around 400 seconds each. All three four-depot D100 instances that exceeded one minute of computing time were optimally solved within less than two minutes. We observed that the the algorithm is, in general, faster for the two-depot instances than for the four-depot instances. Nevertheless, for instances with few vertices and a few connected components, the algorithm is usually faster on the four-depot instances.

With respect to the number of added inequalities, there were considerably more parity cuts than any other type of cuts, even when the number of added cuts was not very large. The number of depot inequalities (15) is, on average, smaller than three. In fact, for only 29 two-depot and 55 four-depot instances was any cut of this type generated. Tables 4 and 5 provide comparisons with the results obtained in [16] for instances with two and four depots, respectively. The method used in [16] is a branch-and-cut algorithm based on a formulation in which the variables depend explicitly on the depot and uses a time limit of one hour per instance, whereas our current branch-and-cut algorithm is based on a formulation in which the variables are associated with traversed edges only. The comparison in terms of computing times is fair since the experiments of this work were performed on the same computer

Table 3: Computational results for instances with four depots

	$\#opt_0$	gap_0	$cutC_0$	$cutP_0$	$cutD_0$	$\#opt$	gap	cutC	cutP	cutD	$\#Nod$	CPU
ALB	0/2	1.14	6.00	110.00	0	2/2	0	4.50	107.00	0	11.5	52.62
P	12/17	0.42	2.35	15.24	0.18	17/17	0	0.18	10.06	0.88	1.81	0.33
D16	6/6	0	0.17	1.17	0	-	-	-	-	-	0	0.00
D36	7/9	0.03	4.00	29.56	1.11	9/9	0	0.44	2.33	0.11	0.22	0.19
D64	3/9	0.86	6.89	57.22	1.33	9/9	0	2.56	44.56	0.56	3.78	4.03
D100	0/9	12.24	12.78	90.56	0.56	9/9	0	7.33	171.22	2.56	14.67	56.30
G16	7/7	0	0.43	3.14	0.71	-	-	-	-	-	0.29	0.01
G36	6/9	0.90	3.22	21.56	2	9/9	0	0.44	4.33	0.11	1.78	0.13
G64	5/9	0.77	3.22	22.22	0.33	9/9	0	1.33	20.33	1.78	11.56	0.95
G100	4/9	1.32	8.44	35.89	1.33	9/9	0	5.33	71.89	1.22	39.67	15.89
R20	2/2	0	0	2	2	-	-	-	-	-	0	0.00
R30	4/5	1.48	0.40	4.40	0.60	5/5	0	0	0	0	0.20	0.01
R40	4/5	0.19	3.60	15.40	0.40	5/5	0	0.40	3.20	1.40	1.60	0.07
R50	2/5	0.68	3.60	21.80	2.80	5/5	0	1.20	4.60	0.60	4.80	0.28

as those of [16]. Each table consists of two blocks of three columns each, the first one for a summary of results from [16] and the second one for a summary of the results with the current branch-and-cut algorithm (referred to as BC). Within each block we present results on the number of instances solved to optimality at the root node ($\#opt_0$), the number of instances optimally solved at termination ($\#opt$) and the total computing time (CPU). As can be seen, our current results notably outperform those of [16]. As mentioned we now solve all instances in less than 400 seconds, whereas in [16] only 95% of the instances were solved to optimality Furthermore, the number of instances optimally solved at the root node is notably larger than that of [16], increasing from 36 to 60 for two-depot instances, and from 53 to 62 for four-depot instances. As can be seen, the computing times are also much smaller. within the time limit of 14,400 seconds. Furthermore, the number of instances optimally solved at the root node is 55% larger than that of [16]. As can be seen, the computing times are also much smaller. The average decreases from 638.39 and 1746.08 seconds for two and four depots, respectively, to 10.99 and 7.90 seconds.

5.1 Numerical results for the new set of larger instances

The good results obtained for small and medium size instances, encouraged us to solve larger instances. Tables 6 and 7 summarize the results obtained with the set of larger two- and four-depot instances with 116 to 744 vertices. Similarly to Table 1, the instances are divided into five groups according their size.

Since formulation (2)–(9) operates on a complete graph, the memory requirements of the solution algorithm after completing the input graph, as described at the beginning of Section 2, become too high when instance size increases. Thus, for

Table 4: Comparison of results with [16] for instances with two depots

	Results from [16]			Results of BC		
	$\#opt_0$	$\#opt$	CPU	$\#opt_0$	$\#popt$	CPU
ALB	0/2	2/2	200.18	1/2	2/2	45.67
P	5/17	17/17	1.87	8/17	17/17	1.01
D16	6/6	-	0.03	3/6	6/6	0.03
D36	1/9	9/9	0.60	7/9	9/9	0.22
D64	0/9	9/9	16.24	6/9	9/9	1.36
D100	0/9	8/9	2452.42	2/9	9/9	104.60
G16	5/7	7/7	0.03	6/7	7/7	0.02
G36	3/9	9/9	0.53	6/9	9/9	0.17
G64	2/9	9/9	156.77	7/9	9/9	0.52
G100	0/9	7/9	4631.05	0/9	9/9	6.67
R20	2/2	-	0.02	2/2	-	0.01
R30	4/5	5/5	0.10	3/5	5/5	0.02
R40	4/5	5/5	0.28	5/5	-	0.06
R50	4/5	5/5	0.17	4/5	5/5	0.06

Table 5: Comparison of results with [16] for instances with four depots

	Results form [16]			Results of BC		
	$\#opt_0$	$\#opt$	CPU	$\#opt_0$	$\#opt$	CPU
ALB	0/2	2/2	5476.70	0/2	2/2	52.62
P	11/17	17/17	44.77	12/17	17/17	0.33
D16	6/6	-	0.01	6/6	-	0.00
D36	4/9	9/9	0.96	7/9	9/9	0.19
D64	1/9	9/9	108.64	3/9	9/9	4.03
D100	0/9	7/9	7085.23	0/9	9/9	56.30
G16	7/7	-	0.01	7/7	-	0.01
G36	5/9	9/9	10.50	6/9	9/9	0.13
G64	5/9	9/9	1835.31	5/9	9/9	0.95
G100	1/9	3/9	9640.11	4/9	9/9	15.89
R20	2/2	-	0.02	2/2	-	0.00
R30	4/5	5/5	0.08	4/5	5/5	0.01
R40	4/5	5/5	0.35	4/5	5/5	0.07
R50	3/5	5/5	0.45	2/5	5/5	0.28

the larger instances in the benchmark sets ALB2, GRP, MAD, URP5 and URP7, after completing the input graph, we remove all unrequired edges $(i, j) \in F$ for which $c_{ij} = c_{ik} + c_{kj}$ for some $k \in V$, and one of two parallel edges whenever they both have the same cost, resulting in a considerable reduction on the total number of edges. Unfortunately, in the resulting (uncomplete) graph optimality condition O2 no longer holds, so MDARP cannot be modeled with binary variables x and y defined above, since non-required edges representing shortest paths between any pair of vertices do not necessarily exist. Fortunately, it is possible to adapt formulation (2)–(9) to that case, with the same meaning for the binary variables x_e and considering general integer y_e variables, whose meaning is now the number of additional traversals of edge $e \in E$. The resulting formulation now reads:

$$\min \sum_{e\in E} c_e(x_e) + \sum_{e\in E^y} c_e(y_e)$$

$$\begin{array}{ll}
(x+y)(\delta(v)) \geq 2 & v \in D \\
(x+y)(\delta(S)) \geq 2 & S \subseteq V \setminus D \\
(x-y)(\delta(S)\setminus H) \geq (x-y)(H) - |H| + 1 & S \subset V, H \subseteq \delta(S), |H| \text{ odd} \\
(x-y)(Q) + (x+y)(Q') \geq (x-y)(H) - |H| + |D'| & S \subset V \setminus D, D' = \{d_i\}_{i\in I} \subset D \\
 & |D'| > 1, H_i \subseteq \delta(S) \cap \delta(d_i), \\
 & |H_i| \text{ odd}, i \in I, H = \bigcup_{i\in I} H_i, \\
 & Q = (\delta(S)\backslash H) \cap \delta(D'), \\
 & Q' = (\delta(S)\backslash H)\backslash \delta(D) \\
x_e = 1 & e \in R \\
y_e \leq 2|D|x_e & e \in E \\
x_e \in \{0,1\} & e \in E \\
y_e \in \mathbb{Z}_+ & e \in E.
\end{array}$$

As can be seen, all 44 instances with up to 500 vertices (ALB2, GRP, MAD, and URP5) are solved to optimality for both two and four depots. For these instances, a provable optimal solution was found at the root node for 19 and 18 instances with two and four depots, respectively.

Two different behaviors can be observed regarding the computational effort required for solve these groups of instances. On the one hand, most of the instances with up to 200 vertices (ALB2, GRP and MAD), are solved within less than two minutes. Only three MAD instances require up to five minutes of computing time, whereas the most time consuming instance requires nearly 30 minutes. On the other hand, the instances of group URP5 require several hours to be solved. The average computing time to solve URP5 instances is around three hours for the two-depot instances and 10 hours for the four-depot instances. There are two and three two-depot instances, which can be solved within less than one and two hours, respectively. The maximum computing time for an instance of this group is around 13 hours. When four-depot instances are considered, only one instance is solved in less than two

Table 6: Computational results for big size instances with two depots

	$\#opt_0$	gap_0	$cutC_0$	$cutP_0$	$cutD_0$	$\#opt$	gap	cutC	cutP	cutD	$\#Nod$	CPU
ALB2	10/14	0.55	13.79	125.57	0	14/14	0	2.64	21.93	0	2.50	11.95
GRP	4/10	0.91	15.50	75.60	0	10/10	0	3.70	31.50	0.10	15.60	6.31
MAD	4/12	0.40	17.08	130.92	0	12/12	0	22.25	166.83	0	15.83	236.31
URP5	1/8	0.53	60.13	442.00	0.88	8/8	0	34.38	447.13	0	17.50	10765.78
URP7	0/8	22.38	83.63	475.50	0	2/8	22.15	50.38	602.63	0	29.75	79011.30

Table 7: Computational results for big size instances with four depots

	$\#opt_0$	gap_0	$cutC_0$	$cutP_0$	$cutD_0$	$\#opt$	gap	cutC	cutP	cutD	$\#Nod$	CPU
ALB2	10/14	0.23	8.14	158.14	0.14	14/14	0	2.00	33.93	0.07	1.36	26.83
GRP	5/10	0.88	12.20	70.90	0	10/10	0	4.90	27.80	0	17.40	5.16
MAD	3/12	0.47	13.92	156.00	0.08	12/12	0	7.00	58.92	0.50	4.00	126.89
URP5	0/8	0.80	53.75	414.00	0.13	8/8	0	59.25	791.00	1.13	75.25	37382.25
URP7	0/8	21.25	84.25	449.38	0	1/8	21.06	44.25	721.25	0	26.00	83741.26

hours, and three of them required more than 12 hours. The maximum computing time is about 21 hours.

Preliminary experiments highlighted the difficulties of solving the URP7 instances as in several cases the algorithm terminated after 24 hours without even finding a feasible solution. Therefore, for these instances, we have implemented a simple heuristic, which allowed us to provide an initial upper bound to the branch-and-cut algorithm. The heuristic consists of two steps. First, to ensure parity, we added to the set of required edges the edges of minimum cost perfect matchings in the subgraphs induced by the odd vertices of each connected component. Then, we add two copies of the edges of T_C, in order to ensure connectivity. However, after the 24 hours limit time, only two instances with two depots (UR732 and UR737) and one with four depots (UR732) were solved to optimality. The algorithm could not find a feasible solution for all the other instances, with the exception of the UR737 with four depots. For this last instance, the gap in the root node was 1.28%, which was reduced to 0.42% at the end. For the other instances, the gap (calculated with the heuristic solution) was nearly 28% either at the root node or at termination, because there was almost no improvement in the value of the lower bound.

6 Conclusions

We have studied the Multi-Depot Rural Postman Problem (MDRPP), which is the extension of the RPP to the case of several depots. A worst-case analysis of the MDRPP with respect to the RPP and other variations indicates that the potential savings can be arbitrarily large, but also that in some cases the one-depot RPP may produce better solutions. A new compact integer linear formulation for the MDRPP containing only binary variables was presented, in which the variables are associated only to the traversed edges. The formulation includes a new family of inequalities that ensure that routes start and end at the same depot. The properties of the polyhedron associated with the formulation were studied. Furthermore, we have developed a branch-and-cut algorithm for the MDRPP based on the proposed formulation. The algorithm is capable of solving to optimality instances involving up to four depots, 744 vertices, 140 required components and 1000 required edges within reasonable computing times, nearly 60% of which at the root node. At termination 96.4% of all instances were solved to optimality.

Acknowledgements

This research was partially supported by the Spanish Ministry of Economy and Competitiveness and EDRF funds through grants EEBB-I-16-10670, BES-2013-063633, and MTM2012-36163-C06-05, MDM-2014-044, and MTM2015-63779-R (MINECO/FEDER), and by the Canadien Natural Sciences and Engineering Research Council under grant 2015-06189. This support is gratefully acknowledged. Thanks are due to the referees for their valuable comments.

References

[1] A. Amberg, W. Domschke, and S. Voß. Multiple center capacitated arc routing problems: A tabu search algorithm using capacitated trees. *European Journal of Operational Research*, 124(2):360–376, 2000.

[2] J. Aráoz, E. Fernández, and C. Franquesa. The clustered prize-collecting arc routing problem. *Transportation Science*, 43(3):287–300, 2009.

[3] J. Aráoz, E. Fernández, and O. Meza. Solving the prize-collecting rural postman problem. *European Journal of Operational Research*, 196(3):886–896, 2009.

[4] J. Aráoz, E. Fernández, and C. Zoltan. Privatized rural postman problems. *Computers & Operations Research*, 33(12):3432–3449, 2006.

[5] F. Barahona and M. Grötschel. On the cycle polytope of a binary matroid. *Journal of Combinatorial Theory, Series B*, 40(1):40–62, 1986.

[6] A. Butsch, J. Kalcsics, and G. Laporte. Districting for arc routing. *INFORMS Journal on Computing*, 26(4):809–824, 2014.

[7] N. Christofides, V. Campos, Á. Corberán, and E. Mota. An algorithm for the rural postman problem. Technical report, Imperial College Report IC.O.R.81.5, 1981.

[8] Á. Corberán, E. Fernández, C. Franquesa, and J.M. Sanchis. The windy clustered prize-collecting arc routing problem. *Transportation Science*, 45(3):317–344, 2011.

[9] Á. Corberán and G. Laporte. Arc Routing: Problems, Methods, and Applications. *MOS-SIAM Series on Optimization, Philadelphia*, 20, 2014.

[10] Á. Corberán, I. Plana, A. Rodríguez-Chía, and J.M. Sanchis. A branch-and-cut algorithm for the maximum benefit Chinese postman problem. *Mathematical Programming*, 141(1):21–48, 2013.

[11] Á. Corberán, I. Plana, and J.M. Sanchis. A branch & cut algorithm for the windy general routing problem. *Networks*, 49(4):245–257, 2007.

[12] Á. Corberán and J.M. Sanchis. A polyhedral approach to the rural postman problem. *European Journal of Operational Research*, 79(1):95–114, 1994.

[13] Á. Corberán and J.M. Sanchis. The general routing problem polyhedron: Facets from the RPP and GTSP polyhedra. *European Journal of Operational Research*, 108(3):538–550, 1998.

[14] E. Fernández, D. Fontana, and M.G. Speranza. On the collaboration uncapacitated arc routing problem. *Computers & Operations Research*, 67:120–131, 2016.

[15] E. Fernández, O. Meza, R.S. Garfinkel, and M. Ortega. On the undirected rural postman problem: Tight bounds based on a new formulation. *Operations Research*, 51(2):281–291, 2003.

[16] E. Fernández and J. Rodríguez-Pereira. The multi-depot rural postman problem. *TOP*, (to appear), 2017.

[17] G. García Ayala, J.L. González-Velarde, R. Ríos-Mercado, and E. Fernández. A novel model for arc territory design: Promoting Eulerian districts. *International Transactions in Operational Research*, 23:433–458, 2015.

[18] G. Ghiani and G. Laporte. A branch-and-cut algorithm for the undirected rural postman problem. *Mathematical Programming*, 87(3):467–481, 2000.

[19] B.L. Golden and R.T. Wong. Capacitated arc routing problems. *Networks*, 11(3):305–315, 1981.

[20] D. Gusfield. Very simple methods for all pairs network flow analysis. *SIAM Journal on Applied Mathematics*, 19(1):143–555, 1993.

[21] A. Hertz, G. Laporte, and P. Nanchen-Hugo. Improvement procedures for the undirected rural postman problem. *INFORMS Journal on Computing*, 11:53–62, 1999.

[22] C. Hierholzer. Über die Mögglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechnung zu umfahren. *Matematische Annalen*, 6:30–32, 1873.

[23] H. Hu, T. Liu, N. Zhao, Y. Zhou, and D. Min. A hybrid genetic algorithm with perturbation for the multi-depot capacitated arc routing problem. *Journal of Applied Sciences*, 13(16):32–39, 2013.

[24] A. Kansou and A. Yassine. A two ant colony approaches for the multi-depot capacitated arc routing problem. In *International Conference on Computers & Industrial Engineering, 2009. CIE 2009.*, pages 1040–1045. IEEE, 2009.

[25] D. Krushinsky and T. Van Woensel. Lower and upper bounds for location-arc routing problems with vehicle capacity constraints. *European Journal of Operational Research*, 244(1):100–109, 2015.

[26] L. Muyldermans. Routing, districting and location for arc traversal problems. Technical report, PhD dissertation, Catholic University of Leuven, Leuven, Belgium, 2003.

[27] L. Muyldermans, D. Cattrysse, and D. Van Oudheusden. Districting for arc-routing applications. *Journal of the Operational Research Society*, 54:1209–1221, 2003.

[28] L. Muyldermans, D. Cattrysse, D. Van Oudheusden, and T. Lotan. Districting for salt spreading operations. *European Journal of Operational Research*, 139(3):521–532, 2002.

[29] L. Muyldermans and G. Pang. Variants of the capacitated arc routing problem. In Á. Corberán and G. Laporte, editors, *Arc Routing: Problems, Methods and Applications*, pages 223–254. MOS-SIAM Series on Optimization, Philadelphia, 2014.

[30] C. S. Orloff. A fundamental problem in vehicle routing. *Networks*, 4(1):35–64, 1974.

[31] S. Wøhlk. Contributions to arc routing. Technical report, PhD dissertation, University of Southern Denmark, Odense, Denmark, 2004.

[32] L. Xing, P. Rohlfshagen, Y. Chen, and X. Yao. An evolutionary approach to the multidepot capacitated arc routing problem. *IEEE Transactions on Evolutionary Computation*, 14(3):356–374, 2010.