New models and algorithms for several families of Arc Routing Problems

Jessica Rodríguez Pereira

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Department of Statistics and Operations Research

New models and algorithms for several families of Arc Routing Problems

Author:
Jessica Rodríguez Pereira

Thesis Advisors:
Elena Fernández Aréizaga
Gilbert Laporte

Barcelona, January 2018

The research carried out in this PhD dissertation has been partially supported by the Spanish Ministry of Economy and Competitiveness and EDRF funds through research projects MTM2012-36163-C06-05 and MTM2015-63779-R (MINECO/FEDER), grants EEBB-I-17-11968, EEBB-I-16-10670, and BES-2013-063633, and by the Canadien Natural Sciences and Engineering Research Council under grant 2015-06189.

To Cito.

Abstract

Some of the most common decisions to be taken within a logistic systems at an operational level are related to the design of the vehicle routes. Vehicle Routing Problems and Arc Routing Problems are well-known families of problems addressing such decisions. Their main difference is whether service demand is located at the vertices or the edges of the operating network.

In this thesis we focus on the study of several arc routing problems. We concentrate on three families of problems. The first family consists of Multi Depot Rural Postman Problems, which are an extension of Rural Postman Problems where there are several depots instead of only one. The second family of problems that we study are Location-Arc Routing Problems, in which the depots are not fixed in advance, so their location becomes part of the decisions of the problem. We finally study Target-Visitation Arc Routing Problems, where the service is subject to an ordering preference among the connected components induced by demand arcs. Different models are studied for each considered family. In particular, two different Multi Depot Rural Postman Problem models are considered, which differ in the objective function: the minimization of the overall transportation cost or the minimization of the makespan. Concerning Location-Arc Routing Problems, we study six alternative models that differ from each other in their objective function, whether there is an upper bound on the number of facilities to be located, or whether there are capacity constraints on the demand that can be served from selected facilities. Finally, two Target-Visitation Arc Routing Problem models are studied, which differ from each other in whether or not it is required that all the required edges in the same component are visited consecutively.

The aim in this thesis is to provide quantitative tools to the decision makers to identify the best choices for the design of the routes. To this end and for each considered problem, we first study and analyze its characteristics and properties. Based on them we develop different Integer Linear Programming formulations suitable for being solved trough branch-and-cut. Finally, all formulations are tested trough extensive computational experience. In this sense, for Multi Depot Rural Postman Problems and Location-Arc Routing Problems we propose natural modeling formulations with three-index variables, where variables are associated with edges and facilities. For some of the models we also present alternative formulations with only two-index variables, which are solely associated with edges. Finally, for the Target-Visitation Arc Routing Problems we propose three different formulations, two alternative formulations for the general case, and one for the clustered version, where all the edges in the same components are served sequentially, which exploits some optimality conditions of the problem.

Resum

Algunes de les decisions més habituals que es prenen en un sistema logístic a nivell operatiu estan relacionades amb el disseny de rutes de vehicles. Els coneguts *Vehicle Routing Problems* i *Arc Routing Problems* són famílies de problemes que s'ocupen d'aquest tipus de decisions. La principal diferència entre ambdós recau en si la demanda de servei es troba localitzada als vèrtexs o a les arestes de la xarxa.

Aquesta tesi es centra en l'estudi de diversos problemes de rutes per arcs. Ens centrem en tres famílies de problemes. La primera família consisteix en els *Multi Depot Rural Postman Problems*, que són una extensió del *Rural Postman Problem* on hi ha diversos dipòsits en lloc d'un de sol. La segona família de problemes que estudiem són els *Location-Arc Routing Problems*, en els quals els dipòsits no estan fixats amb antelació i, per tant, la seva ubicació esdevé part de les decisions a prendre en el problema. Finalment, estudiem els *Target-Visitation Arc Routing Problems*, on el servei està subjecte a una preferència d'ordenació entre les components connexes induïdes pels arcs amb demanda. S'estudien diferents models per a cadascuna de les famílies considerades. En particular, es consideren dos models diferents per al *Multi Depot Rural Postman Problem*, que es diferencien en la funció objectiu: la minimització del cost general de transport o la minimització de la ruta més llarga. Pel que fa als *Location-Arc Routing Problems*, estudiem sis models alternatius que difereixen en la seva funció objectiu, considerant si hi ha un límit màxim sobre la quantitat de dipòsits a ubicar o si hi ha restriccions de capacitat sobre la demanda que es pot servir des dels dipòsits seleccionats. Finalment, s'estudien dos models de *Target-Visitation Arc Routing Problem*, que es diferencien en si es necessari que totes les arestes requerides en la mateixa component es visitin de forma consecutiva.

L'objectiu d'aquesta tesi és proporcionar eines quantitatives als responsables, que permetin identificar les millors opcions de disseny de les rutes. Per això, i per a cadascun dels problemes considerats, primer estudiem i analitzem les seves característiques i propietats. A partir d'aquestes, desenvolupem diferents formulacions de Programació Lineal Entera, adequades per a la seva solució mitjançant un *branch-and-cut*. Finalment, totes les formulacions són provades mitjançant un ampli testeig computacional. En aquest sentit, per als *Multi Depot Rural Postman Problems* i els *Location-Arc Routing Problems*, proposem formulacions naturals amb variables de tres índexs, on les variables estan associades a les arestes i als dipòsits. Per a alguns dels models també presentem formulacions alternatives, amb variables de només dos índexs, que només estan associades a les arestes. Finalment, per als *Target-Visitation Arc Routing Problems* proposem tres formulacions diferents, dues formulacions alternatives per al cas general i una per a la versió en clúster, on totes les arestes de la mateixa component es serveixen seqüencialment, cosa que explora algunes condicions d'optimització pròpies.

Resumen

Algunas de las decisiones más habituales que se toman en un sistema logístico a nivel operativo están relacionadas con el diseño de rutas de vehículos. Los conocidos *Vehicle Routing Problems* y *Arc Routing Problems* son familias de problemas que se ocupan de este tipo de decisiones. La principal diferencia entre ambas reside en si la demanda de servicios está localizada en los vértices o en las aristas de la red.

Esta tesis se centra en el estudio de diversos problemas de rutas por arcos. Nos centramos en tres familias de problemas. La primera familia consiste en los *Multi Depot Rural Postman Problems*, que son una extensión del *Rural Postman Problem* donde hay varios depósitos en lugar de solamente uno. La segunda familia de problemas que estudiamos son los *Location-Arc Routing Problems*, en los que los depósitos no están fijados con antelación y, por lo tanto, su ubicación se convierte en parte de las decisiones a tomar en el problema. Finalmente, estudiamos los *Target-Visitation Arc Routing Problems*, donde el servicio está sujeto a una preferencia de ordenación entre las componentes conexas inducidas por los arcos con demanda. Se estudian diferentes modelos para cada una de las familias consideradas. En particular, se consideren dos modelos diferentes para el *Multi Depot Rural Postman Problem* que se diferencian en la función objetivo: la minimización del coste general de transporte o la minimización de la ruta más larga. En cuanto a los *Location-Arc Routing Problems*, estudiamos seis modelos alternativos que difieren en su función objetivo, en si hay un ímite máximo sobre la cantidad de depósitos a ubicar, o en si hay restricciones de capacidad sobre la demanda que se puede servir desde los depósitos seleccionados. Finalmente, se estudian dos modelos de *Target-Visitation Arc Routing Problem*, que se diferencian en si es necesario que todas las aristas requeridas en la misma componente se visiten de forma consecutiva.

El objetivo de esta tesis es proporcionar herramientas cuantitativas a los responsables, que permitan identificar las mejores opciones de diseño de las rutas. Por ello, y para cada uno de los problemas considerados, primero estudiamos y analizamos sus características y propiedades. A partir de estas, desarrollamos diferentes formulaciones de Programación Lineal Entera, adecuadas para su solución mediante un *branch-and-cut*. Finalmente, todas las formulaciones son probadas mediante un amplio testeo computacional. En este sentido, para los *Multi Depot Rural Postman Problems* y los *Location-Arc Routing Problems*, proponemos formulaciones naturales con variables de tres índices, donde las variables están asociadas a las aristas y a los depósitos. Para algunos de los modelos también presentamos formulaciones alternativas con variables de sólo dos índices, que sólo están asociadas a las aristas. Finalmente, para los *Target-Visitation Arc Routing Problems* proponemos tres formulaciones diferentes, dos formulaciones alternativas para el caso general y una para la versión en clúster, donde todas las aristas de la misma componente se sirven secuencialmente, lo que explora algunas condiciones de optimización propias.

Acknowledgements

After an intensive period, writing this note of thanks is the finishing touch on my thesis. It has been a period of learning for me, not only in the scientific area, but also on a personal level. Along the development of this thesis, I have had the pleasure of working with some researchers from whom I learnt a lot. I would like to reflect on the people who have supported and helped me so much throughout this period.

Firstly, I would like to express my sincere gratitude to Elena Fernández who has introduced me to the operation research world and has helped me along all the way by offering me her unwavering patience, guidance, support and expertise. I am also specially grateful to Gilbert Laporte for the continuous support to my research, for his patience, encouragement, and immense knowledge and for giving me the opportunity to develop part of this work in excellent conditions, during my stay in the Centre Interuniversitaire de Recherche sur les Reseaux d'Entreprise, la Logistique et le Transport.

I would like to extend my thanks to Gerhard Reinelt for his support and suggestions in the third part of the thesis, and for giving me the opportunity to develop part of my thesis in a very relaxed environment in the Universität Heidelberg.

I would like to acknowledge the warm hospitality, all the facilities and help I received from all the members of the Departament d'Estadística i Investigació Operativa of the Universitat Politècnica de Catalunya where most of this thesis has been developed. I would like to also extend this gratitude to all members of the CIRRELT and the University of Heidelberg where I spent my research stages.

I would like to thank my fellow postgraduate students for the support, friendship, and for all the fun we have had in the last four years. Specially, thanks to Cris, Valentina and Vicky. I would also like to thank all the research colleagues with whom I have met in different events, courses, or congresses. They have also been part of the development of this work, as well as my professional and personal growth.

Last but not the least, I would like to express my gratitude to my family and my friends for their unconditional support and encouragement in the difficult moments.

Contents

List of Figures

List of Tables

Preface

Currently, the management of logistic systems has become a crucial aspect for the efficient organization of industrial companies and service delivery. More and more, the aggressive competition in national and international markets has led companies to realize the need of adopting integrated logistic systems in order to survive successfully. Such systems allow to achieve a greater degree of efficiency and competitive advantage over potential competitors, since they reduce costs, save operational time, and optimize resources and processes. In its turn, such savings imply a positive impact on the quality of the customers services, the brand image, and the corporate reputation.

In this context, the staff responsible for the management of logistics systems must take decisions at strategic, tactical, and operational levels, involving long, medium, and short term financial impact respectively. Even if these decisions depend on the nature of the company, most often they involve location and sizing of facilities, materials management, production planning, order processing, inventory management, storage systems, transport system and return management.

Operations Research is a versatile discipline that provides quantitative tools to the decision makers to identify the best choices. It is within this framework that in this thesis we apply operation research techniques to study some decisions concerning logistic systems, in particular, those related to the design of routes.

Routing problems define one of the main classes of problems arising at tactical or operational levels within logistics. The choice of the routes for vehicles involve the assignment of users to facilities and/or vehicles. Given a set of demand customers, a set of open facilities and a fleet of vehicles with a specific structure, routes must be established for attending the existing demand, in order to optimize the selected optimization criterion, which often considers the minimization of transportation costs. The family of problems that address such issues is known as Vehicle Routing Problems or Arc Routing Problems depending on whether the service demand is located at the vertices or edges of the network's graph.

In this thesis, we focus on three families of Arc Routing Problems. In particular, we study Multi Depot Rural Postman Problems, Location-Arc Routing Problems, and Target-Visitation Arc Routing Problems. The motivation for studying these problems comes not only from their theoretical interest, but also from their potential applications. Similarly to other arc routing problems, such applications appear in a wide variety of practical cases. Mail delivery, garbage collection, road maintenance, snow plowing or pipelines inspection are typical examples of real-life applications.

One the one hand, when considering large application areas for Arc Routing Problems, 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. For instance, in urban waste collection companies usually operate from multiple depots. If the depot location are known in advance, a possibility for handling such problems is to decompose them in as many independent problems as depots, first allocating to the different depots demand sectors within smaller operating areas, and then finding optimal routes within each sector. Such solution strategy is indeed suboptimal, as it can be possible to obtain better solutions if a global approach is applied in which the allocation and routing decisions are jointly addressed. Therefore, an integrated joint approach is preferable, motivating the study of Multi Depot Rural Postman Problems. Furthermore, the selection of the locations for the depots is an strategic logistic decision, with an important impact on the above mentioned tactical decisions. This suggests studying Location-Arc Routing Problems to jointly address the location of facilities (depots) and the design of service routes. On the other hand, the study of Target-Visitation Arc Routing Problems is motivated by some practical applications in which service demand is subjected to preferences concerning the order in which edges with demand have to be traversed. For instance, in snow plowing or natural disasters where it may be preferable to serve the demand of some edges or clusters before others.

For the different classes of Arc Routing Problems that we study, we focus on modeling aspects, suitable formulations, and efficient solution methods. Different models are studied for each considered family. In particular, two alternative Multi Depot Rural Postman Problems models are considered, which differ in their objective function: the minimization of the overall transportation cost or the minimization of the makespan. Concerning Location-Arc Routing Problems, we study six alternative models that differ from each other in their objective function, whether there is an upper bound on the number of facilities to be located, or whether there are capacity constraints on the demand that can be served from selected facilities. Finally, two Target-Visitation Arc Routing Problem models are studied, which differ from each other in whether or not it is required that all the required edges in the same component are visited consecutively.

For Multi Depot Rural Postman Problems we propose natural modeling formulations with three-index variables, where variables are associated with edges and the facilities associated with the routes that traverse them. An alternative formulation with only two-index variables, associated with edges but not with facilities, is presented for the model that minimizes the overall costs. This approach indeed reduces the number of required variables at the expense of presenting some additional difficulties for suitably defining the routes. To this end, we introduce a new set of constraints guaranteeing that the routes are consistent and return to the original depot. Three and two-index formulations are also proposed for the different Location-Arc Routing Problems, taking into account that the two-index formulation is only valid for models in which the objective is an aggregate measure of all routes, and the feasibility of the solucions can be derived form the aggregate information. Finally, for the Target-Visitation Arc Routing Problems we propose three diferent formulations, two alternative formulations for the general case, and one for the clustered version, where all the edges in the same components are served sequentially, which exploits some optimality conditions of the problem.

All proposed formulations are suitable for being solved with branch-and-cut algorithms. In each case, we present exact separation algorithms for the families of inequalities of exponential size. The resulting algorithms have been implemented and computationally tested. The obtained results are presented and analyzed.

In the following chapters we describe in detail the studied problems, as well as the proposed formulations and corresponding solution methods. The structure of this thesis is as follows. In Chapter 1 we introduce some basic concepts related to the developments of this thesis. An overview of the existing literature related to the three families of studied Arc Routing Problems is presented in Chapter 2. Next, we define the Multi Depot Rural Postman Problems under study and their properties in Chapter 3, where a disaggregate formulation is proposed and a branch-and-cut is presented, together with the obtained results. In Chapter 4 an alternative compact formulation for the min-cost Multi Depot Rural Postman Problem is proposed, as well as its polyhedral analysis, a branch-and-cut solution, and the results of extensive computational experience. The formal definition of the studied Location-Arc Routing Problems and their solution methods are developed in Chapter 5, where six alternative models are studied. In Chapter 6 we present two Target-Visitation Arc Routing Problems variants together with a preliminary computational experience. We conclude this thesis with some remarks in Chapter 7.

Some of the results of this thesis have been published in journals or presented at conferences or workshops. The publications and conference participation are listed below:

Publications

- E. Fernández, G. Laporte, and J. Rodríguez-Pereira. *Exact Solution of Several Families of Location-Arc Routing Problems*. Submitted.
- E. Fernández, G. Laporte, and J. Rodríguez-Pereira. *A branch-and-cut algorithm for the multi-depot rural postman problem*, in Transportation Science, forcoming 2018.
- E. Fernández and J. Rodríguez-Pereira. *Multi-depot rural postman problems*, in TOP 25, 2017.

Conferences

- J. Rodríguez-Pereira, E. Fernández and G. Laporte. *Formulations for Location-Arc Routing Problems* in IFORS-Conference of the International Federation of Operational Research Societies, 2017. Québec, Canada.
- J. Rodríguez-Pereira, E. Fernández and G. Laporte. *A Branch-and-Cut algorithm for the Multi-Depot Rural Postman Problem* in Middle-European Conference on Applied Theoretical Computer Science (MATCOS), 2016. Koper, Slovenia.
- J. Rodríguez-Pereira, E. Fernández and G. Laporte. *Formulation and exact solution of the Multi-Depot Rural Postman Problem* in The second Workshop on Arc Routing Problems (WARP), 2016. Lisbon, Portugal.
- J. Rodríguez-Pereira, E. Fernández and G. Laporte. *Undirected Multi-Depot Rural Postman Problem* in Optimization Days (OPTD), 2016. Montreal, Canada.
- J. Rodríguez-Pereira and E. Fernández. *Multi-Depot Rural Postman Problem*, in The fifth Workshop on Combinatorial Optimization, Routing and Location (CORAL), 2015. Salamanca, Spain.
- J. Rodríguez-Pereira and E. Fernández. *Multi-Depot Rural Postman Problem*, in The XXXV Congreso Nacional de Estadística e Investigación Operativa (SEIO), 2015. Pamplona, Spain.

Chapter 1

Preliminaries

1.1 Graphs

In this section we introduce the notation and conventions that are used in this thesis, related to graphs.

An *undirected graph* $G = (V, E)$ is a pair of two finite sets: the set of *vertices*, $V = \{1, 2, ..., n\}$, and the set of *edges*, $E \subseteq \{\{i, j\} | i, j \in V; i < j\}$. The graph is called *complete* when there is an edge connecting every pair of vertices, $E = \{\{i, j\} \in V \times V | i < j\}$. An edge e is *incident* to a vertex v, if v is one of the end-vertices of e. The two vertices that define an edge are said to be *adjacent*. The number of edges incident to a given vertex v or, equivalently, number of vertices adjacent to v is referred to as the *degree* of v, $|\delta(v)|$. The vertex v is called *even* or *odd* if its degree is, respectively, even or odd.

Given an undirected graph G, a *path* connecting v_0 and v_l is a sequence of edges from vertex v_0 to vertex v_l of the form $\{\{v_0, v_1\}, \{v_1, v_2\}, ..., \{v_{l-1}, v_l\}\}$. The particular case in which the first and last vertices of the path are the same, $v_0 = v_l$, defines a closed path known as *tour*, *route*, or *cycle*. If the tour visits exactly once each edge it is called *Eulerian tour*, otherwise, if the tour visits at least once each edge of set of edges, it is called a *postman tour*. A graph G is *connected* when there is a path between every pair of vertices. When G is not connected, each connected subgraph defines a connected component. A *forest* is a subset of edges that no contains any cycle. Furthermore, if the subgraph is connected, then is known as *tree*.

For a non-empty subset of vertices $S \subseteq V$ we denote by $\gamma(S) = \{\{u, v\} \in E | u, v \in S\}$ the set of edges with both vertices in S, and by $\delta(S) = \{\{u, v\} \in E | u \in S, v \notin S\}$, the set of edges with one vertex in S and the other vertex in $\overline{S} = V \setminus S$, referred, also as edges of the *cut* of S. In addition, we will use the following usual notation. 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 $f(A) \equiv f(A \cap B) \equiv \sum_{e \in A \cap B} f_e$.

A *directed* graph will be denoted by $N = (V, A)$, where vertices are usually called *nodes* and there is a set of *arcs*, A, with pairs of ordered vertices, $a = (i, j)$. To differentiate from the case of an undirected graph where edges are unordered pairs of vertices, we use the convention of representing an arc $a = (i, j)$ as an ordered pair of nodes, where i is the *starting* node and j the *end* node. Note that now the arc (i, j) is different from the arc (j, i). The set of arcs with v as starting node is the set of *arcs outgoing* from v and is denoted by $\delta^-(v)$. Analogously, the set of arcs with v as end-node is the set of *incoming arcs* of v, $\delta^+(v)$. In a directed graph the degree of a vertex is divided into *outdegree* and *indegree* depending on the number of outcoming and incoming arcs in the vertex, respectively. All concepts above for undirected graphs can be extended to directed graphs through a few natural adaptations.

1.2 Combinatorial Optimization

Combinatorial optimization (CO) is a broad area that deals with the study of optimization problems in which feasible solutions can be expressed as combinations of elements of a finite set. CO has multiple applications in operations research, software engineering, artificial intelligence, etc. In general, CO problems are NP-hard (Wolsey and Nemhauser, 2014), so the enumeration of all feasible solutions is unsuitable. There exist however several methodologies for dealing with CO problems. Often CO problems can be stated as problems on graphs so graph theory concepts can be used. Moreover, most CO problems can be formulated as 0-1 Linear Programs. Then techniques from discrete optimization and, in particular, from integer programming can be applied to address them. The basic theory of integer programming will be introduced below.

1.2.1 Polyhedral combinatorics

In this section we summarize the main concepts of polyhedral combinatorics. Most of this material has been taken from Grötschel and Padberg (1985).

Let $x_1, ..., x_k \in \mathbb{R}^n$ and $\lambda_1, ..., \lambda_k \in \mathbb{R}$, the vector $x \in \mathbb{R}$ with $x = \lambda_1 x_1 + ... + \lambda_k x_k$ is called *linear combination* of vectors $x_1, ..., x_k$. Furthermore, λ satisfies $\lambda_1 + ... + \lambda_k = 1$, then x is an *affine combination* of the vectors $x_1, ..., x_k$.

A *hyperplane* is defined by the set $\{x \in \mathbb{R}^n | ax = a_0\}$, while the set $\{x \in \mathbb{R}^n | ax \leq a_0\}$ defines a *halfspace*. The intersection of finitely many halfspaces characterizes a *polyhedron* P as $P = \{x \in \mathbb{R}^n | Ax \leq b\}$. This is also know as *H-representation* or the *outer description*. The polyhedron P is called *polytope* when it can be described by a convex hull, i.e. $P = Conv(X)$. The *dimension* (denoted by dim) of a polyhedron or polytope is the maximum number of affinely independent points minus one. So, a polytope P is *full-dimensional* if $P \subseteq \mathbb{R}^n$ and $dimP = n$

An inequality $ax = a_0$ is *valid* for a polytope P if the inequality is satisfied by all the points in P. The set $f = P \cap ax = a_0$, for a valid inequality $ax = a_0$, is a *face* of the polytope. When a face contains one element only ($dimf = 0$) it is called a *vertex*. Otherwise, a nonempty face which is maximal with respect to set inclusion ($dimf = dimP - 1$) is named a *facet*. The inequality associated with f is known as *face-defining* or *facet-defining*, respectively. Two valid inequalities, $\{a^T x \leq a_0\}$ and $\{b^T x \leq b_0\}$, are *equivalent* if they define the same face, i.e. $\{x \in P | a^T x = a_0\} = \{x \in P | b^T x = b_0\}$. However, if exists $\lambda > 0$ that verifies $a^T > \lambda b^T$ and $a_0 \leq \lambda b_0 \leq$ then we said that $a^T x = a_0$ *dominates* $b^T x = b_0$.

1.2.2 Integer Programming

Below we summarize some concepts of integer programming. Most of this material has been taken from Wolsey (1998).

Integer Programming is one of the most relevant fields in operations research and, in particular, in mathematical optimization. Its main goal is to solve optimization problems with discrete or integer variables of the form:

$$\begin{aligned} (IP)Min \quad & cx \\ \text{s.t.} \quad & Ax \geq b \\ & x \geq 0 \text{ and integer} \end{aligned} \tag{1.1}$$

where c is a row vector in $\mathbb{R}^n$, x is a vector of decision *variables*, A is a coefficient matrix of dimension (m, n), and b is a independent term vector in $\mathbb{R}^m$. The linear function cx is the *objective function* and the inequalities $Ax \geq b$ conform the m *constraints* of the problem. The set of solutions that satisfies the constraints, $F := \{x \in \mathbb{Z}^n_+ | Ax \geq b\}$ is known as the *feasible* set of solutions. A feasible solution x^* is an *optimal solution* if and only if $cx^* \leq cx \; \forall x \in F$. Note that there may be *alternative* optimal solutions if there are more than one $x \in F$ with $cx = cx^*$. A Binary Integer Program (BIP) is a special case of integer program where integer variables may only take the values 0 or 1. Such variables are known as binary variables.

A problem (RP) $z^R = min\{fx : x \in T \subseteq R^n\}$ is a *relaxation* of an IP $z = min\{cx : x \in X \subseteq R^n\}$ if it has a larger set of feasible solutions, $X \subseteq T$, and the objective function has the same or a smaller value everywhere, $fx \leq cx$ for all $x \in X$. One of the most useful and natural relaxations is the *linear programming relaxation*. For an IP $min\{cx : x \in P \cap Z^n\}$, with $P = \{x \in R^n_+ : Ax \geq b\}$, the linear programming relaxation (LP) is the linear program $z^{LP} = min\{cx : x \in P\}$.

Solving integer programs requires sophisticated algorithms, as *Branch-and-Bound*, the *cutting planes*, and *Branch-and-Cut* (see 1.3).

1.3 Solution methods

The techniques for solving linear integer programs can be classified in exact and heuristic methods. Many heuristic methods can be found in the literature. In general, such methods do not guarantee the optimality of the obtained solutions and are not further described in this section. The interested reader is addressed to Wolsey (1998). Most exact algorithms, some of which are described next, are based on enumeration procedures, where non-optimal solutions are eliminated by bounding procedures. In general, these algorithms do not have polynomial complexity, but may work efficiently in practice.

1.3.1 Branch-and-bound

Broadly speaking branch-and-bound consists in splitting the problem into smaller subproblems, which are solved or split again. This methodology is typically represented by means of a search tree where the original problem is the root node and each subproblem is represented by a node in the tree.

Branch-and-bound is a useful method for solving IPs. For that, the first step is to solve the LP relaxation of the IP. If the solution to the LP has all the variables with integer values, the solution obtained is optimal for the original IP. Otherwise, some variable, x_i, which has a fractional value, v, is selected to create two new problems by adding a new constraint. In the first one, we add the constraint $x_i \leq \lfloor v \rfloor$. In the second problem we add the constraint $x_i \geq \lceil v \rceil$. Then the new problems are recursively solved until all subproblems are examined. When a subproblem has an integer solution, the value of the solution is kept as the incumbent, if it is better (lower) than the current best integer solution. The value of the incumbent is used to discard subproblems whose linear relaxation solution is equal or worse (greater) than the incumbent value, reducing the number of subproblems to explore, at the same time as it reduces the enumeration procedure.

The advantage of branch-and-bound lies in its simplicity. However it is important to obtain good global upper bounds in short time, which allows to build, in general, a smaller exploration tree than the tree produced by complete enumeration. There are some methods to obtain good bounds, such as applying heuristics or using a proper selection criteria of the subproblem, to find a solution close to the optimum.

1.3.2 Cutting Planes

Cutting planes are broadly used for solving integer programs. The method operates on a domain containing all feasible points, which is iteratively redefined by adding linear inequalities, called cuts. In many cases a family of inequalities contains an exponential number of inequalities, so all of them cannot be added a priori. In such a case it is better to add them iteratively. For this, the first step is to solve the LP relaxation of the given IP and to obtain a solution x^*. This solution is tested for being an integer. As long x^* is not be integer, a separation problem for x^* is solved. If a violated inequality that separates x^* from the convex hull is found, then this added to the current LP and resolved. The process continues until the solution is fully integral.

The crucial point on this algorithm is to be able to apply cuts that approximate the convex hull of the feasible set as much as possible. Thus, the best inequalities that can be included are those defining facets of the polytope of the original problem. In general, the knowledge about the studied optimization problem and its associated polytope allows to obtain good inequalities.

1.3.3 Branch-and-cut

Branch-and-cut is commonly used to solve NP-hard combinatorial optimization problems. This technique combines branch-and-bound with the generation of cutting planes to tighten the IP relaxations. Thus, a branch-and-cut algorithm is a branch-and-bound algorithm in which cutting planes are generated throughout the branch-and-bound-tree. A branch-and-cut algorithm first computes the LP relaxation of the IP. At that point, if the solution has a non-integer value for some variable, a cutting plane phase is applied, and violated or valid inequalities are added to the linear program to reinforce the relaxation, as long as some violated or valid inequality is found. Then, two new subproblems are defined according to the branching phase. The algorithm goes on, successively, in each new subproblem.

1.4 Arc Routing Problems

Vehicle routing is a widely studied area within CO widely studied, which consists in designing routes for a fleet of vehicles to serve a set of customers with a demand to satisfy. According to Corberán and Laporte (2014), Arc Routing Problems (ARPs) are characterized by the fact that service demand is placed at the edges or arcs of a given network, instead of at the nodes. These problems present a wide variety of applications such as mail and newspaper delivery, waste collection, snow plowing, salt spreading, meter reading, lines inspection, school bus route, and other pick up, delivery or services problems through the links of a network.

The origin of this research field goes back to the 18^{th} century with the Königsberg bridges problem, but it is not until the late sixties of the 20^{th} century that it begins to be a strong field of interest for research groups. Over the past years several variants of problems have been studied. However, three basic arc routing problems can be distinguished:

Chinese Postman Problem (CPP), consists in finding the minimum cost tour that passes through each edge of a given graph at least once.

Rural Postman Problem (RPP), extends the CPP to serve only a required subset of edges, while the rest are only traversed if it is necessary to define the tour.

Capacitated Arc Routing Problem (CARP), generalizes the previous problems, with a fleet of vehicles that should satisfy the demand of the required subset of edges, bearing in mind the limited capacity of vehicles.

In the following we want to focus on the RPP, which is essential for this thesis, since the studied problems are extensions of it. For more details of ARPs, we refer the interested reader to Corberán and Laporte (2014), Dror (2000), Eiselt et al. (1995a,b)

1.4.1 The Rural Postman Problem

The RPP is defined on an undirected graph $G = (V, E)$, where V is the vertex set, $|V| = n$ and E is the edge set, $|E| = m$. We denote by $R \subset E$ the set of required edges that must be traversed, and by $F = E \setminus R$ the set of unrequired edges. Each edge $e \in E$ is associated with non-negative real cost c_e. The connected components induced by the required edges are referred to as *required components* and 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 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 $C_k, k \in K$. V_C contains a node representing each connected component C_k, $k \in K$. For each pair of distinct components C_k and $C_{k'}$, E_C contains an edge $\{k_e, k'_e\}$ associated with each original edge e linking C_k and $C_{k'}$, i.e. each edge $e \in \delta_F(V_k) \cap \delta_F(V_{k'})$, which inherits its cost from G.

The RPP consists of determining a minimum cost tour traversing all required edges at least once, or equivalently, determining a least cost set of deadhead edges which, together with the required edges yields a tour. This NP-hard problem was introduced by Orloff (1974) and its complexity was proven by Orloff (1976). However, it may be solved in a polynomial time when the graph induced by the required edges, R is connected, as it can be reduced to the undirected CPP.

Usually the RPP is addressed on a transformed graph in which only the required vertices, V_R, are taken into account. To this end, the procedure described in Christofides et al. (1981) is as follows. First, an edge between every pair of vertices of V_R is added to $G_R = (V_R, R)$, having a cost equal to the shortest path length on G. Then, all unrequired edges $\{i, j\} \in F$ for which $c_{ij} = c_{ik} + c_{kj}$ for some $k \in V$, are removed from G, as well as one of two parallel edges whenever they both have the same cost. Hence the costs of the simplified graph satisfy the triangle inequality.

1.4.1.1 Optimality conditions

Optimal RPP solutions satisfy a series of properties, which have been proposed by Christofides et al. (1981), Corberán and Sanchis (1994), Ghiani and Laporte (2000):

(O1) (Christofides et al., 1981). Every optimal solution satisfies that the maximum number of additional copies of an edge e to obtain an Eulerian graph is 1 if $e \in R$, or 2 otherwise, $e \in F$.

(O2) (Corberán and Sanchis, 1994). Let $e = \{i, j\}$ be an edge such its adjacent vertices, i and j, belong to some connected component C_k. Then, in every optimal solution, the number of additional copies of e is equal to 1. Note that for edges $e \in R$ the outcome is equivalent to previous one.

(O3) (Ghiani and Laporte, 2000). Let $e^{(1)}, e^{(2)}, ..., e^{(l)}$ be the edges having exactly one vertex in a given component C_h and one vertex in another given component C_k $(h, k \in \{1, ..., K\}, h \neq k)$. In an optimal solution, only and edge $e^{(r)}$ such $c(e^{(r)}) = min\{c(e^{(1)}), c(e^{(2)}), ..., c(e^{(l)})\}$ can be traversed twice. Consequently, no more than $K(K-1)/2$ edges need to be traversed twice in an optimal solution.

(O4) (Ghiani and Laporte, 2000). There exists an optimal solution in which at most $K - 1$ edges are traversed twice.

(O5) (Ghiani and Laporte, 2000). The only edges $e \in F$ that can be traversed twice in an optimal solution, are those belonging to the T_C.

1.4.1.2 Mathematical formulation

There are several formulations for the RPP. However, it is worth mentioning the first pure binary integer lineal program introduced by Ghiani and Laporte (2000). The variable x_e represents the number of additional copies of edge $e \in E$ that must be added so that together with the set of required edges a tour is defined. Let E_{012} (E_{01}) be the set of edges that can be traversed at most twice (once) in an optimal solution. For each edge $e \in E_{012}$ the authors define two binary variables x'_e and x''_e. Let E' (E'') be the set of edges e' (e''), and let $\bar{\mathrm{E}}= E_{01} \cup E' \cup E''$. Then, the formulation is as follows:

$$\text{minimize} \sum_{e \in \bar{\mathrm{E}}} c_e x_e \tag{1.2}$$

subject to

$$x(\delta(S)) \geq 2 \qquad S = \bigcup_{k \subset K} V_k,\ k \neq \emptyset \tag{1.3}$$

$$x(\delta(i) \setminus H) \geq x(H) - |H| + 1 \qquad i \in V_R,\ H \subseteq \delta(i),\ |H| \text{ odd if } i \text{ is } R\text{-even},\ |H| \text{ even if } i \text{ is } R\text{-odd} \tag{1.4}$$

$$x_e = x'_e + x''_e \qquad e \in E_{012} \tag{1.5}$$

$$x'_e, x''_e \in \{0,1\} \qquad e \in E_{012} \tag{1.6}$$

$$x_e \in \{0,1\} \qquad e \in \bar{\text{E}}. \tag{1.7}$$

In this formulation, constrains (1.3) force the solution graph to be connected, while constraints (1.4) ensure the degree of each vertex to be even, by imposing that, if an odd (even) number of edges $e \in H$ are incident to a R-even (R-odd) vertex $i \in V_R$, then at least another edge has to be incident to vertex i. The autors also presented an extension of the parity inequalities (also called cocircuit inequalities) to non-empty subset $S \subset V_R$:

$$x(\delta(S) \setminus H) \geq x(H) - |H| + 1 \qquad S \subset V_R,\ H \subseteq \delta(S),\ |H| \text{ odd if } S \text{ is } R\text{-even},\ |H| \text{ even if } S \text{ is } R\text{-odd} \tag{1.8}$$

Next we rewrite the above formulation with the notation that will be used throughout this thesis. For this we slightly modify the meaning and the domain of the decision variables. In particular, two sets of binary variables are considered, associated with the first and second traversal of edges, respectively. We denote by $E^y \subset E$ the set of edges that can be traversed twice in an optimal solution. E^y consists of demand edges, R, plus the edges of T_C, (corresponding to $R \cup E_{012}$). For each $e \in E$, let x_e be a binary variable indicating whether or not edge e is traversed. For each $e \in E^y$, let y_e be a binary variable that takes the value one if and only if edge e is traversed twice. Then formulation (1.2)–(1.7) becomes:

$$\text{minimize} \sum_{e \in E} c_e x_e + \sum_{e \in E_y} c_e y_e \tag{1.9}$$

subject to

$$(x+y)(\delta(S)) \geq 2 \qquad S \subset V \tag{1.10}$$

$$(x-y)(\delta(S) \setminus H) + y(H) \geq x(H) - |H| + 1 \qquad S \subset V,\ H \subseteq \delta(S),\ |H| \text{ odd} \tag{1.11}$$

$$x_e = 1 \qquad e \in R \tag{1.12}$$

$$y_e \leq x_e \qquad e \in E^y \tag{1.13}$$

$$x_e \in \{0,1\} \qquad e \in E \tag{1.14}$$

$$y_e \in \{0,1\} \qquad e \in E^y. \tag{1.15}$$

Similarly to the previous formulation, constraints (1.10) force the solution to be connected and cocircuit constraints (1.11) that each vertex or set $S \subset V$ has an even degree. Furthermore, it is now necessary to ensure that all required edges are traversed (1.12). Finally, the condition that the second traversal can not be used unless the first traversal has been used is imposed in (1.13).

1.4.1.3 Separation of connectivity and parity constraints

The size of the families of connectivity and parity constraints (1.10) and (1.11), respectively, is exponential in $|V|$. Thus, only a small subset can be included in the initial formulation, whereas the remaining inequalities are only incorporated when violated by the solution to the current relaxation. As we next see, for both families of inequalities the separation problem can be solved exactly. Throughout, $(\overline{x}, \overline{y})$ denotes the current LP solution. Furthermore, $G_{\overline{x},\overline{y}} = (V_{\overline{x},\overline{y}}, E_{\overline{x},\overline{y}})$ denotes the support graph of the solution, obtained from G by eliminating all edges in E with $\overline{x}_e = 0$ and all vertices that are not incident with any edge of $E_{\overline{x},\overline{y}}$.

Connectivity inequalities (1.10) The separation for inequalities (1.10) is to find a set $S \subset V \setminus D$, with $(\overline{x} + \overline{y})(\delta(S)) \geq 2$, or to prove that no such inequality exists. Violated connectivity constraints are associated with minimum cuts in $G_{\overline{x},\overline{y}}$ relative to the capacities vector $\overline{x} + \overline{y}$. Thus, solving the separation problem of (1.10) consists of building the tree of min-cuts T (see, for instance, Gusfield, 1993) of $G_{\overline{x},\overline{y}}$ relative to $\overline{x} + \overline{y}$ and of identifying each min-cut $\delta(S)$ of T of value $v^{V \setminus D}(S) < 2$. The inequality (1.10) associated with such a cut-set $\delta(S)$ is violated by the current solution $(\overline{x}, \overline{y})$.

It is possible to apply a heuristic separation of (1.10) looking for connected components in the graph $G_{\overline{x},\overline{y}}$, 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, where $V_C \subset V \setminus D$. If $(\overline{x} + \overline{y})\,(\delta(V(C))) < 2$, the connectivity inequality associated with $V(C)$ is violated by $(\overline{x}, \overline{y})$.

Parity inequalities (1.11) The separation problem for inequalities (1.11) is to find $S \subset V$, $H \subseteq \delta(S)$, $|H|$ odd such that

$$(\overline{x} - \overline{y})(\delta(S) \setminus H) + \overline{y}(H) < \overline{x}(H) - |H| + 1, \tag{1.16}$$

or to prove that no such inequality exists. The procedure that we describe below was introduced for other arc routing problems with binary variables by Aráoz et al. (2009b).

Note that (1.16) can be written as

$$(\overline{x} - \overline{y})(\delta(S) \setminus H) + |H| - (\overline{x} - \overline{y})(H) < 1. \tag{1.17}$$

or equivalently as

$$\sum_{e\in\delta(S)\setminus H}(\overline{x}_e-\overline{y}_e)+\sum_{e\in H}(1-(\overline{x}_e-\overline{y}_e))<1. \qquad (1.18)$$

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 (1.17) 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 (1.17) 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 (1.17) 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, together with S, defines an inequality (1.11) violated by the current solution $(\overline{x},\overline{y})$.

Thus, the separation problem can be solved by finding the tree T 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 (1.11) after making H odd is not necessarily associated with the smallest min-cut of the tree. When T has a cut $\delta(S)$ of capacity smaller than one, i.e. $b(\delta(S))<1$, then the vertex set S, and the set of edges $H=\{e\in\delta(S)\mid(\overline{x}_e-\overline{y}_e)\geq 0.5\}$ are considered. If $|H|$ is odd, H defines, together with S, a violated inequality of type (1.11). Otherwise, if $|H|$ is even, H can be transformed to an odd set, as explained, by either removing one edge from H (and transferring it to $\delta(S)\setminus H$) or by adding to H one edge currently in $\delta(S)\setminus H$. Then, when $b(\delta(S))+\Delta<1$, the updated set H defines a violated inequality (1.11) for S in the current solution $(\overline{x},\overline{y})$.

A heuristic procedure for the above separation problem 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 (1.11) associated with S and H is violated by $(\overline{x},\overline{y})$. Otherwise, if $|H|$ is odd and $b(\delta(S))+\Delta\geq 1$, then the heuristic fails.

Chapter 2

Literature review

The nature of ARPs leads to a variety of related problems, some of which will be developed over the course of this thesis. In this chapter we review the most relevant literature related to the problems under study. It is structured in three sections corresponding to the three types of problems that we have studied. In Section 2.1 we present an overview of Multi Depot Arc Routing Problems (MDARPs), in which the objective is to minimize the overall routing costs or to minimize the makespan. Section 2.2 is devoted to Location-Arc Routing Problems (LARPs), which combine decisions for the location of multiple facilities with arc routing decisions. Finally, Section 2.3 focuses on Target-Visitation Arc Routing Problems (TVARPs), which incorporate ordering preferences to the RPP.

2.1 Multi-Depot Arc Routing Postman Problems

MDARPs are the extension of ARPs, to the case when there are several depots instead of only one. The literature on MDARPs is scarce. To the best of our knowledge, the only existing exact algorithms for the Multi Depot Rural Postman Problem (MDRPP) have been developed in Fernández et al. (2016) and, in relation to this thesis, in Fernández and Rodríguez-Pereira (2017), and Fernández et al. (2018). A directed MDARP dealing with carriers collaboration is considered in Fernández et al. (2016), where an exact branch-and-cut algorithm is developed for a collaborative arc routing problem solving to optimality instances with up to 50 vertices and two depots. Fernández et al. (2018), and Fernández and Rodríguez-Pereira (2017) propose exact algorithms to solve the MDRPP on an undirected graph. A branch-and-cut based on a binary linear formulation is proposed in Fernández and Rodríguez-Pereira (2017). The formulation uses *natural* decision variables, which explicitly indicate the depot with which each traversed edge is associated. On the contrary, Fernández et al. (2018) use decision variables associated only with edges, but not with the facilities. In Fernández and Rodríguez-Pereira (2017) instances with up to 100 vertices and four depots are solved to optimality, whereas in Fernández et al. (2018) instances involving up to 744 vertices and four depots are solved to optimality.

Other than this, all previous work on MDARPs we are aware of has focused on multi-depot capacitated arc routing problems (MDCARPs). Some theoretical aspects of MDCARPs are considered in Wøhlk (2004). A new formulation and exact solution algorithm are presented in Krushinsky and Van Woensel (2015) for the asymmetric MDCARP. Heuristics methods have been put forward for both the undirected and the directed MDCARP. Sequential heuristics for the undirected MDCARP are proposed in Amberg et al. (2000), Muyldermans et al. (2002), and Muyldermans et al. (2003). A cluster-first-route-second strategy, where the assignment of arcs to depots is established before designing the routes is applied in Amberg et al. (2000), and a route-first-cluster-second strategy is used in Muyldermans et al. (2002), and Muyldermans et al. (2003), 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 Kansou and Yassine (2009), and a hybrid genetic algorithm with perturbation that incorporates a local search, a replacement method, and a perturbation mechanism is proposed in Hu et al. (2013). The directed case is addressed in Xing et al. (2010), where an evolutionary approach is presented, which takes advantage of the extensions of the heuristics for the classical single-depot CARP Golden and Wong (1981).

Multi depot routing problems are indeed related to districting, where a set of clusters or districts that suitably partition the demand set is sought. The design of good districts, which takes place at a strategic level, where demand points or edges are allocated to facilities, allows finding efficient routes in each district at an operational 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 Muyldermans et al. (2002, 2003) are devised as a second phases in districting design problems. Two recent works on districting for arc routing are Butsch et al. (2014), and García Ayala et al. (2015). The interested reader is addressed to Muyldermans (2003), Muyldermans and Pang (2014) for further reading on this topic.

The MM-MDRPP deals with the minimization of the makespan. This min-max objective has been notably less studied than the usual minimization of the overall routing cost. The min-max objective was introduced by Frederickson et al. (1976) for several arc and node routing problems to obtain balanced routes. The MM-MDRPP is related to the the min-max K-Rural Postman Problem (MM-K-RPP), which is a particular case of the MM-MDRPP, by considering K depots co-located at the same vertex. The MM-K-RPP is an uncapacitated arc routing problem, with one single depot and a fixed number of vehicles, K. Each vehicle must perform a tour starting and ending at the facility. The objective is to minimize the length of the longest among the K routes. This has been studied by several authors for different types of graphs

Benavent et al. (2009, 2010, 2011, 2013, 2014), Willemse and Joubert (2012).

2.2 Location-Arc Routing Problems

LARPs combine location and routing decisions in contexts where some arcs of a given network must be serviced. LARPs were formally introduced by Ghiani (1998), but an earlier publication by Levy and Bodin (1989) describes an application in the United States Postal Service in which a postman parks his van in several locations from which he proceeds to deliver mail on foot.

LARPs are the arc routing counterpart of Location Routing Problems (LRPs) arising in node routing contexts (see Albareda-Sambola, 2015, Drexl et al., 2013, Min et al., 1998, Nagy and Salhi, 2007, Prodhon and Prins, 2014, for surveys), but have ben less extensively studied. According to Albareda-Sambola (2015), this may be due to the fact that ARPs can often be transformed into node routing problems, as in Baldacci and Maniezzo (2006), Longo et al. (2006), Pearn et al. (1987). To the best of our knowledge, Ghiani and Laporte (1999) and Arbib et al. (2014) presented the only exact algorithms for uncapacitated LARPs. Ghiani and Laporte (1999) reduce the original problem to an undirected RPP and solve it by means of an exact branch-and-cut algorithm. Arbib et al. (2014) present a mathematical programming formulation and a branch-and-cut algorithm for a directed profitable LARP in which the facilities are located at both endpoints of the selected arcs according to the facility opening costs, to the profit collected on these arcs, and to the cost of traversing them.

Some authors have focused on capacitated LARPs. Hashemi Doulabi and Seifi (2013) present two formulations on mixed graphs: one for the general case, and one for the case of a single facility. They propose a simulated annealing heuristic, which incorporated several arc routing heuristics. Lopes et al. (2014) present a four-index flow formulation as well as constructive, classical improvement heuristics and metaheuristics. Several authors have studied extensions of the CARP with a location component. In Ghiani et al. (2001) location decisions are related to intermediate facilities at which vehicles such as garbage trucks can unload in order not to exceed their capacity. Pia and Filippi (2006) consider a CARP with mobile depots, Amaya et al. (2007) solve a CARP in which extra vehicles replenish the main fleet at meeting points to be located. The authors formulate the problem and solve it by means of a cutting plane algorithm. Salazar-Aguilar et al. (2013) study a related problem in the context of road marking.

2.3 Target-Visitation Arc Routing Problems

TVARPs, are the arc routing counterpart of Target Visitation Problems (TVPs), which combine routing and the Linear Ordering Problem (LOP) (see Martí and

Reinelt, 2011). In TVARPs the targets are associated with the edges of the network instead of with the nodes.

The literature on TVPs is limited since they are a relatively new problems. These problems were introduced by Grundel and Jeffcoat (2004) for planning optimal routes for unmanned aerial vehicles in military missions. Hildenbrandt and Reinelt (2015), and Hungerländer (2015) present exact algorithms for the TVPs. In Hildenbrandt and Reinelt (2015) a branch-and-cut algorithm is developed for solving the problem to optimality, while in Hungerländer (2015) an exact semidefinite optimization approach is proposed. Other than this, all previous work on TVPs we are aware of has focused on heuristic methods. Arulselvan et al. (2007) propose two genetic algorithms, one focusing on a hybrid genetic algorithm and the second one dealing with a random key genetic algorithm. Blázsik et al. (2006) also propose some heuristics methods. To the best of our knowledge there is no research besides these mentioned papers, especially in what respects to the arc routing version.

TVARPs are connected to ARPs with hierarchies on the set of arcs, where the order in which the demand must be visited is established in advance. The Hierarchical CPP (HCPP) was introduced by Dror et al. (1987) and studied by other authors Cabral et al. (2004), Ghiani and Improta (2000), Korteweg and Volgenant (2006). Another related problem is the RPP with deadline classes which has been studied by Letchford and Eglese (1998). The HCPP and the RPP with deadlines consider a single-vehicle arc routing problem in which the required edges are partitioned into a number of classes according to priorities, each class having its own order or deadline, respectively. More recent works of this related problems are Colombi et al. (2016) and Colombi et al. (2017).

Chapter 3

Multi-Depot Rural Postman Problems

In this chapter we present MDRPPs on undirected graphs. Similarly to other arc routing problems, in MDRPPs service demand is placed at a subset of edges. The distinguished feature of MDRPPs is that there are several depots instead of just one. MDRPPs involve two types of decisions: the allocation of the demand edges to the facilities and the construction of the set of routes. Thus, feasible solutions are given by sets of routes, each of them starting and ending at one of the depots, where each demand edge is traversed at least once by some route.

We consider two different MDRPP models, which differ from each other in the objective function. The first model uses a min-cost objective where the goal is to determine a set of routes of minimum total cost and will be referred to as MC-MDRPP. The MC-MDRPP extends to several depots the well-known undirected RPP, which considers one single facility.

The second model that we study uses a min-max objective where the goal is to minimize the makespan, that is the length of the longest route, and will be referred to as MM-MDRPP. In contrast to the MC-MDRPP, which minimizes the overall routing costs, but may produce routes, which are unbalanced in terms of their length, the MM-MDRPP can be suitable when balanced routes are sought. The MM-MDRPP is related to the MM-K-RPP, which considers K facilities co-located at the same vertex.

For each of the models that we study we introduce a Mixed Integer Linear Programming (MILP) formulation, referred to as tree-index or disaggregate, where variables are associated with edges and depots. Moreover, a branch-and-cut algorithm has been implemented and tested on a large set of benchmark instances. Most of the content of this chapter has been published in Fernández and Rodríguez-Pereira (2017).

3.1 Formal definition

MDRPPs are defined on an undirected connected graph $G = (V, E)$, where V is the vertex set, $|V| = n$, and E is the edge set, $|E| = m$. 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. Like in the RPP, the connected components induced by the required edges are referred to as *required components* and denoted by $C_k = (V_k, R_k)$, $k \in K$, so $R = \bigcup_{k \in K} R_k$, and $V_R = \bigcup_{k \in K} V_k$. Let c be a non-negative real cost function defined on the edges of G. Again we denote by T_C the MST with respect to cost function c, of the multigraph $G_C = (V_C, E_C)$ induced by the connected components. Unless otherwise stated, we assume that G has been simplified like in the RPP (see Chapter 1).

We use the term *route* to denote a closed path, not necessarily simple, that starts and ends at the same facility $d \in D$. When the facility associated with the route needs to be explicit we say that the route is *rooted* at depot d. We say that a required edge $e \in R$ is *served* by a route, if the route traverses e 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 in the route.

Definition 3.1.1.

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

- *The MM-MDRPP is to find a set of routes, one from each depot, that serve all the required edges and minimize the length of the longest route.*

3.1.1 Modeling assumptions

In our study of the MDRPP we assume with respect to the set of depots D, 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 $k \in K$. Therefore, in no case an edge connecting two depots belongs to the set of required edges, R. Note that this assumption implies that a route rooted at depot d does not traverse any other depot different from d.

Regarding the service of demand, we consider that required edges in the same component can be served from different facilities. The effect of this assumption is illustrated in Figure 3.1.a that shows the input graph, which has two required components with one depot in each of them (v_1 and v_2, respectively). Black lines represent required edges, while unrequired edges are drawn in light grey. The numbers next to the edges indicate their costs. Figure 3.1.b shows the optimal solution when it is imposed that all required edges in the same component are served from the same depot, of cost $z = 23$. The route of v_1 (represented with solid lines), which serves the demand of the required edges of C_1, consists of edges {v_1, A}, {A, B}, and {v_1, B}. The route rooted at depot v_2 (represented with dotted lines), which serves the required edges of C_2, consists of edges {v_2, E}, {C, E}, {C, D}, {D, E}, {E, F}, and {v_2, F}. Figure 3.1.c shows that a better solution of cost $z = 19$ can be obtained if we allow to split the components and serve required edges in the same component from different facilities. Now all the required edges of C_1 and some required edges of C_2 are served in the route rooted at depot v_1 defined by edges {v_1, A}, {A, C}, {C, E}, {D, E}, {B, D}, and {v_1, B}. The remaining required edges of component C_2 are served in the route rooted at depot v_2, which consists of edges {v_2, E}, {E, F}, and {v_2, F}.

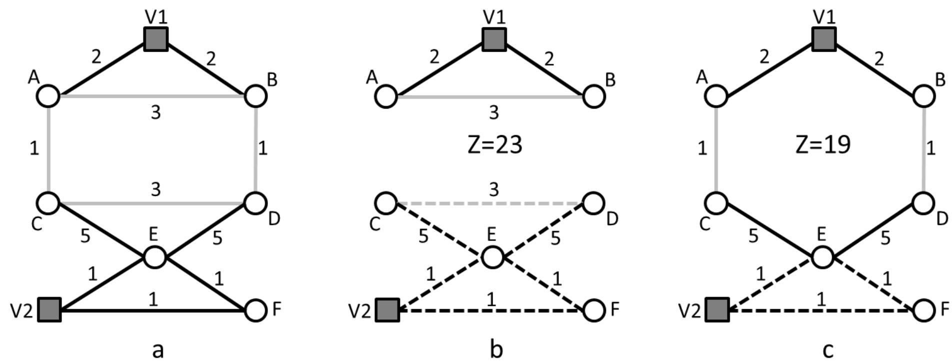


Figure 3.1: Example that allowing to split demand components among routes may produce better solutions

Note that, as a consequence of this modeling assumption, feasible routes are not necessarily vertex-disjoint.

3.1.2 Complexity and optimality conditions

The MC-MDRPP and the MM-MDRPP are NP-hard. It is easy to see that the RPP is a particular case of both problems when $|D| = 1$. Thus, since the RPP is an NP-hard problem (Orloff, 1976), also the MC-MDRPP an the MM-MDRPP belong to the same class.

As it is usual in other related uncapacitated arc routing problems on undirected graphs, the feasibility of MDRPP solutions is basically established via connectivity and parity conditions, in addition to the requirement that all demand edges are served. Thus it is not surprising that, similarly to other such problems with min-cost objectives, when non-negative costs satisfy the triangle inequality, optimality conditions can be extended or adapted for the MC-MDRPP, considering the properties of each route. These optimality conditions will alow us to derive formulations using binary variables only. Furthermore, these properties can also be used or readapted for the MM-MDRPP.

(O1) **MC-MDRPP and MM-MDRPP.** There exists an optimal solution in which no edge is traversed more than twice in each route. Otherwise, two copies of the same edge can be removed without affecting neither the requirement that all demand edges are served, nor the parity of the vertices or the connectivity with the depot.

(O2) **MC-MDRPP and MM-MDRPP.** There exists an optimal solution where no non-demand edge with the two end-nodes in the same component ($e \in \gamma_F(V_k)$) is traversed more than once in each route. Otherwise, two copies of such an edge can be removed without affecting the feasibility of the solution. Furthermore, because of the triangle inequality, the only edges of $\gamma_F(V_k)$, that are used, are those connecting two R-odd vertices.

(O3) There exists an optimal solution in which the only non-demand edges that are traversed twice in the same route are of one of the following types:

(a) **MC-MDRPP.** Edges of T_C.

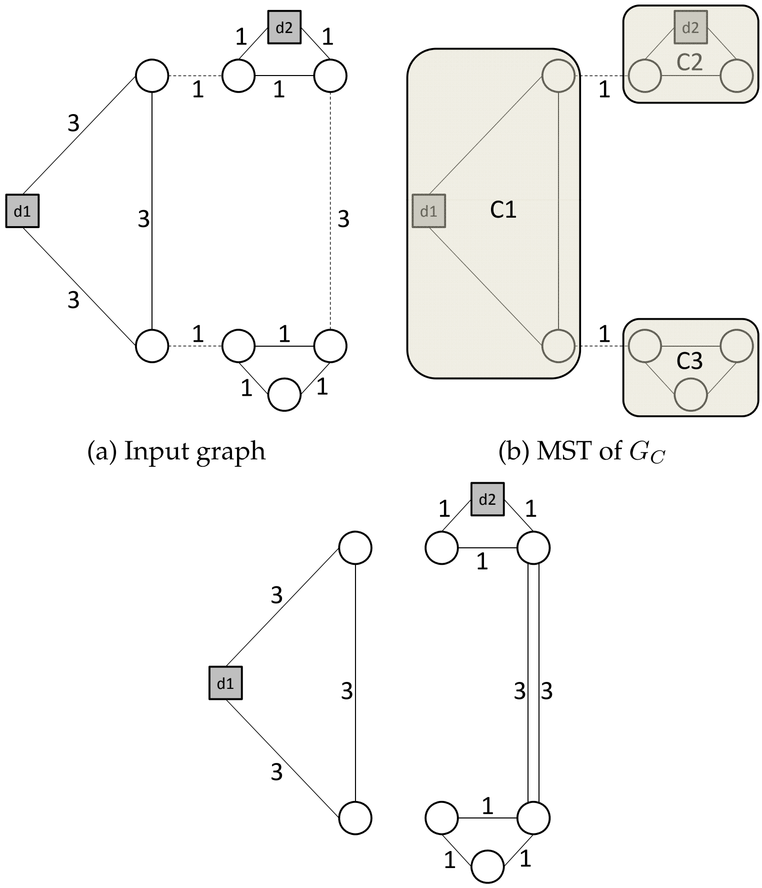
It is clear that any MST of G_C will use only least cost edges between pairs of components. Let T^* be an MST of G_C, and suppose an edge $e^* \in E$ connecting components C_k and $C_{k'}$ is traversed twice in an optimal MC-MDRPP solution s^*, but $\{k_{e^*}, k'_{e^*}\}$ is not a least cost edge of G_C connecting nodes C_k and $C_{k'}$. Then, adding edge e^* to T^* produces a cycle in G_C, in which $c_{\widehat{e}} < c_{e^*}$, where $\widehat{e}$ denotes a least cost edge in such cycle. In this case replacing in s^* the two copies of edge e^* by two copies of $\widehat{e}$ produces a feasible solution: the parity of the vertices of the original graph G does not change and the connectivity of the new solution is guaranteed by the two copies of $\widehat{e}$. It is possible that in the new solution some edges are served from a different facility than in the original solution s^*, but this does not affect to its feasibility either. The fact that the cost of the new solution is smaller than that of the original one, contradicts the optimality of the original solution.

(b) **MM-MDRPP.** Least cost edges connecting pairs of vertices of the multigraph graph $G_C = (V_C, E_C)$.

As shown in the example of Figure 3.2 the edges of T_C are not enough in the case of the MM-MDRPP, where an optimal solution may have two copies of a non-demand edge connecting two different components, which does not belong to T_C. Thus the adaptation of this condition to the MM-MDRPP must take into account all least cost edges connecting any pair of components.

(O4) **MC-MDRPP and MM-MDRPP.** There exists an optimal MDRPP solution in wich each demand edge is served by exactly one route.

(O5) **MC-MDRPP and MM-MDRPP.** There exists an optimal solution in which no edge is traversed more than the number of possible traversals in each route times the number of depots. That is, $2|D|$ times for the required edges as well as for the edges that satisfy O3, and $|D|$ times for all other edges.



(a) Input graph (b) MST of G_C

(c) Optimal solution

Figure 3.2: Optimal MM-MDRP solution using twice an edge not in T_C.

We note that the above conditions are valid for problems stated on any undirected connected graph. Stronger optimality conditions can be derived for the case when G=(V, E) is a complete input graph. In that case all shortest paths connecting each pair of vertices are represented by some edge of E, so multiple traversals of edges can be *hidden* by such edges. This implies that the number of times that each edge can be traversed in an optimal solution is independent of the number of depots. In particular:

(O6) **MC-MDRPP and MM-MDRPP.** If G=(V, E) is a complete graph, there exists an optimal solution in which no edge is traversed more than twice.

(O7) **MC-MDRPP and MM-MDRPP.** If G=(V, E) is a complete graph, there exists an optimal solution in which the only edges that are traversed twice are required edges connecting two R-odd vertices plus edges that satisfy O3.

3.2 Worst-case analysis

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

The costs savings that can be obtained with the MC-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 $z^*(MC-MDRPP)$ is the sum of the optimal Chinese Postman solution values on each component. Figure 3.3 illustrates one such example, which we will use in the proof of Theorem 3.2.1.

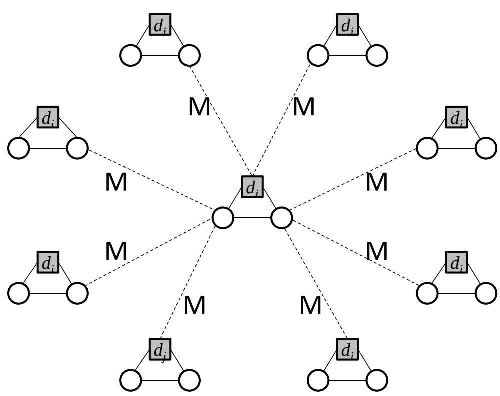


Figure 3.3: Potential savings of the MC-MDRPP relative to the RPP

Theorem 3.2.1. *There exists no finite bound for the ratio* $z^*(RPP)/z^*(MC-MDRPP)$.

Proof. Consider an MC-MDRPP instance like the one depicted in Figure 3.3, defined on 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 MC-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

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

which tends to ∞ when $M \to \infty$. ■

Despite the above result, it is also possible that $z^*(MC-MDRPP)$ 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 $z^*(RPP)/z^*(MC-MDRPP)$.

Theorem 3.2.2. $z^*(RPP)/z^*(MC-MDRPP) \geq 1/2$, *and the bound is asymptotically tight.*

Proof. To see that $z^*(MC-MDRPP) \leq 2z^*(RPP)$ we observe that a feasible solution for a given MC-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.

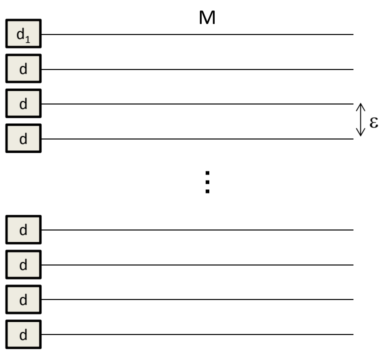
(iii) For each component containing no depot, retain one dipath connecting it with some component with a depot, and eliminate all remaining such dipaths if they exist.

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

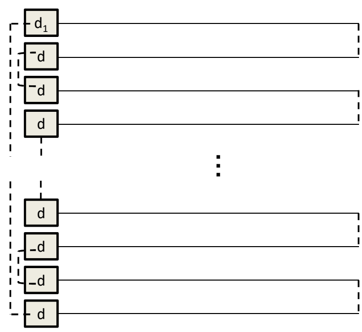
Consider an MC-MDRPP instance like the one depicted in Figure 3.4a defined on a graph $G = (V, E)$, where each required component consists of one single edge represented by a solid line, $R_k = \{\{u_k, v_k\}\}$ of cost M, and contains a depot (gray square) located at its leftmost vertex u_k. Suppose 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. $F = \{\{u_k, u_{k+1}\}\colon 1 \leq k < |K|\} \cup \{\{v_k, v_{k+1}\}\colon 1 \leq k < |K|\}$. Let us finally suppose that the cost of each unrequired edge is ε. In the optimal MC-MDRPP solution to the above instance, each required edge is traversed twice and no unrequired edge is used. Hence, $z^*(MC - MDRPP) = 2|K|M$. The optimal RPP solution (see Figure 3.4b) 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

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

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



(a) Input graph



(b) Optimal RPP solution

Figure 3.4: Potential savings of the RPP relative to the MC-MDRPP.

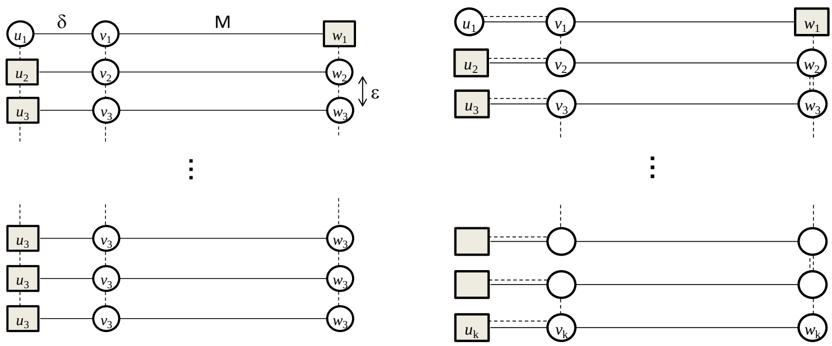
We conclude this section by comparing the value of the MC-MDRPP, in which required edges in the same component can be served from different depots, with its clustered version MC-MDRPP$_C$, in which it is imposed that all the required edges in the same component are served from the same depot. Denote by $z^*(MC - MDRPP_C)$ the optimal value to a given MC-MDRPP$_C$ instance. Since any feasible solution to the MC-MDRPP$_C$ is feasible for the MC-MDRPP, we have that $z^*(MC - MDRPP) \leq z^*(MC - MDRPP_C)$. Thus a lower bound for the ratio $z^*(MC - MDRPP_C)/z^*(MC - MDRPP)$ 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 $z^*(MC - MDRPP_C)/z^*(MC - MDRPP)$ and shows that the bound is asymptotically tight.

Theorem 3.2.3. *$z^*(MC - MDRPP_C)/z^*(MC - MDRPP) \leq 2$, and the bound is asymptotically tight.*

Proof. To see that $z^*(MC - MDRPP_C) \leq 2z^*(MC - MDRPP)$ it is sufficient to observe that replicating all the edges in an optimal MC-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 MC-MDRPP$_C$.

The example of Figure 3.5a 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. $F = \big\{\{u_k, u_{k+1}\}\colon 1 \leq k < |K|\big\} \cup \big\{\{v_k, v_{k+1}\}\colon 1 \leq k < |K|\big\} \cup \big\{\{w_k, w_{k+1}\}\colon 1 \leq k < |K|\big\}$. The cost of each unrequired edge is ε.

An optimal solution to MC-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 $z^*(MC - MDRPP_C) = 2|K|(\delta + M)$.



(a) Input graph (b) Optimal MDRPP solution

Figure 3.5: Potential savings due to splitting the demand of components.

Figure 3.5b depicts an optimal MC-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, $z^*(MC - MDRPP) = 2|K|\delta + (|K|M + 2(|K| - 1)\varepsilon)$. Therefore, for the instance described above we have

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

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

3.3 Three-index formulations

The natural modeling option when dealing with routing problems with multiple depots is to make use of binary variables that associate arcs or edges with facilities, and then define the routes of each one. This offers two main advantages. On the one hand, in absence of capacity or other type of constraints, the feasibility of a route associated with a given depot is guaranteed through the imposition of connectivity and parity 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. Below we present a formulation for each of the proposed models.

3.3.1 Formulation for the Min-cost Multi-Depot RPP

The disaggregate MILP formulation for the MC-MDRPP exploits the optimality conditions. Condition O1 implies that we only need two sets of binary variables, associated with each depot, for 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 MC-MDRPP solution, which consists of all demand edges plus the edges of the MST of G_C (see condition O3a). In each set, variables are associated with the depot of the route that traverses the edges. Hence, the sets of binary decision variables are the following:

For all $e \in E$, $d \in D$,

$$x_e^d = \begin{cases} 1 & \text{if edge } e \text{ is traversed by the route rooted at depot } d \\ 0 & \text{otherwise.} \end{cases}$$

For all $e \in E^y$, $d \in D$,

$$y_e^d = \begin{cases} 1 & \text{if edge } e \text{ is traversed twice by the route rooted at depot } d \\ 0 & \text{otherwise.} \end{cases}$$

Then, a MILP for the MC-MDRPP is as follows:

$$\text{minimize} \sum_{d\in D}\left(\sum_{e\in E} c_e x_e^d + \sum_{e\in E^y} y_e^d\right) \tag{3.1}$$

subject to

$$(x^d + y^d)(\delta(d)) \geq 2 \qquad d \in D \qquad (3.2)$$

$$(x^d + y^d)(\delta(S)) \geq 2x_e^d \qquad d \in D,\ S \subseteq V \setminus \{d\},\ e \in \gamma(S) \qquad (3.3)$$

$$(x^d - y^d)(\delta(S) \setminus H) + y^d(H) \geq x^d(H) - |H| + 1 \qquad S \subset V,\ H \subseteq \delta(S),\ |H| \text{ odd},\ d \in D \qquad (3.4)$$

$$\sum_{d \in D} x_e^d = 1 \qquad e \in R \qquad (3.5)$$

$$y_e^d \leq \sum_{d \in D} x_e^d \qquad e \in E^y,\ d \in D \qquad (3.6)$$

$$x_e^d \in \{0, 1\} \qquad e \in E,\ d \in D \qquad (3.7)$$

$$y_e^d \in \{0, 1\} \qquad e \in E^y,\ d \in D. \qquad (3.8)$$

Inequalities (3.2) and (3.3) ensure that all facilities are used and the connectivity of each route with its associated depot, respectively. This later condition is imposed by stating that if an edge is traversed by the route associated with depot $d \in D$, then at least two edges must cross the cut-set of any vertex set containing its two end-nodes, and not containing the depot d. Inequalities (3.4) ensure the parity of every subset of vertices and, in particular, at every vertex. Broadly speaking, they impose that if a solution uses an odd number of edges, H, 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 (3.6). Thus, 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 (3.4) are an adaptation to the MDRPPs of those proposed in Aráoz et al. (2006), Aráoz et al. (2009a), Aráoz et al. (2009b), for the Prize-collecting and the Clustered RPP, which were later reinforced in Corberán et al. (2013) for the Maximum Benefit CPP. Inequalities (3.2)–(3.4) jointly guarantee that any solution defines $|D|$ Eulerian circuits. Taking into account optimality condition O4, equalities (3.5) ensure that each required edge is served by one route. As mentioned, inequalities (3.6) impose that a route cannot traverse an edge for a second time unless the edge has been traversed for the first time. Binary conditions on the x and y variables, derived from their definition are reflected by constraints (3.7) and (3.8).

The above formulation has $|E| \times |D|$ x variables and $|E^y| \times |D|$ y variables. There are $|D|$ inequalities of type (3.2), $|R|$ inequalities (3.5) and $|E^y| \times |D|$ inequalities of type (3.6). The size of the families inequalities (3.3) and (3.4) is exponential in $|V|$.

3.3.2 Formulation for the Min-Max Multi-Depot RPP

Similarly to the above formulation for the MC-MDRPP, the MILP formulation for the MM-MDRPP uses the same sets of decision binary variables x and

y, taking into account that the index set E^y for the variables associated with edges that can be traversed twice in an optimal MM-MDRPP solution now must be defined according to condition O3b. In addition, we define a new integer variable w representing the length of the longest route, so the objective becomes the minimization of w. The formulation inherits all constraints (3.2)–(3.8), and includes a new family of constraints that relate the new variable w to the lengths of the longest route (3.10).

The MM-MDRPP formulation is as follows:

$$\text{minimize } w \tag{3.9}$$

subject to

$$(x^d + y^d)(\delta(d)) \geq 2 \qquad d \in D \tag{3.2}$$

$$(x^d + y^d)(\delta(S)) \geq 2x_e^d \qquad d \in D,\ S \subseteq V \setminus \{d\},\ e \in \gamma(S) \tag{3.3}$$

$$(x^d - y^d)(\delta(S) \setminus H) + y^d(H) \geq x^d(H) - |H| + 1 \qquad S \subset V,\ H \subseteq \delta(S),\ |H| \text{ odd},\ d \in D \tag{3.4}$$

$$\sum_{d \in D} x_e^d = 1 \qquad e \in R \tag{3.5}$$

$$y_e^d \leq \sum_{d \in D} x_e^d \qquad e \in E^y,\ d \in D \tag{3.6}$$

$$w \geq \sum_{e \in E} c_e x_e^d + \sum_{e \in E^y} c_e y_e^d \qquad v \in D \tag{3.10}$$

$$x_e^d \in \{0, 1\} \qquad e \in E,\ d \in D \tag{3.7}$$

$$y_e^d \in \{0, 1\} \qquad e \in E^y,\ d \in D \tag{3.8}$$

$$w \in \mathbb{Z}^+. \tag{3.11}$$

3.3.3 Valid inequalities

Below we present some families of simple valid inequalities that we will use to reinforce the LP relaxations of the above formulations for the MC-MDRPP and the MM-MDRPP:

1. **Aggregate connectivity constraints.** By adding up over all depots, the connectivity constraints (3.3) associated with subsets of nodes containing no depots, and taking into account that all vertices, except possibly the depots, are incident with some demand edge, and thus must be visited, we obtain:

$$\sum_{d \in D} (x^d + y^d)(\delta(S)) \geq 2 \quad S \subset V, S \cap D = \emptyset. \tag{3.12}$$

Even if the family (3.12) is implied by the general families (3.3) and the family (3.12) is also of exponential size, some small sub-families associated with particular subsets S can be very useful to reinforce the initial LP relaxation when the general family (3.3) is relaxed:

- Singletons $S = \{v\}$, with $v \in V \setminus D$:

$$\sum_{d \in D}(x^d + y^d)(\delta(v)) \geq 2 \quad v \in V \setminus D. \tag{3.13}$$

  For singletons $\{v\} = \{d\}$ with $d \in D$, corresponding to depots, the inequalities (3.13) are also valid, although they are dominated by the stronger constraints (3.2).

- End-nodes of demand edges. $S^e = \{u, v\}$, with $e = \{u, v\} \in R$, $S^e \cap D = \emptyset$:

$$\sum_{d \in D}(x^d + y^d)(\delta(S^e)) \geq 2 \quad e \in R,\ S^e \cap D = \emptyset. \tag{3.14}$$

- Vertex sets of connected components without depot, $S = V_k$, $k \in K$, $V_k \cap D = \emptyset$:

$$\sum_{d \in D}(x^d + y^d)(\delta(V_k)) \geq 2 \quad k \in K,\ V_k \cap D = \emptyset. \tag{3.15}$$

2. **Aggregate parity constraints.** Aggregate versions of the parity constraints (3.4) are indeed valid for the MC-MDRPP and the MM-MDRPP. Similar inequalities (but combining binary and general integer variables) have been used for other ARPs with multiple vehicles, namely the MM-K-RPP Benavent et al. (2009, 2014). For the MC-MDRPP and the MM-MDRPP, for $S \subset V$, $H \subseteq \delta(S)$, $|H|$ odd, the inequality that we obtain is the following:

$$\sum_{d \in D}(x^d - y^d)(\delta(S) \setminus H) + \sum_{d \in D} y^d(H) \geq \sum_{d \in D} x^d(H) - |H| + 1 \tag{3.16}$$

In particular, when S is R-odd, i.e. $|\delta_R(S)|$ odd, and $H = \delta_R(S)$, the inequality (3.16) becomes

$$\sum_{d \in D}(x^d - y^d)(\delta_F(S)) + \sum_{d \in D} y^d(\delta_R(S)) \geq 1. \tag{3.17}$$

3.4 Branch-and-cut Algorithm

We have implemented branch-and-cut solution algorithms, for the MDRPPs defined in Section 3.1, based on the formulations proposed above. The families of constraints of exponential size are relaxed and, at each iteration, inequalities violated by the current LP solution are separated. Such inequalities are iteratively incorporated to the current formulation and the reinforced formulation resolved. For both formulations the two families of constrains of exponential size that are initially relaxed are the connectivity and the parity constraints (3.3) and (3.4), respectively. Below we give the details of the algorithm. In particular, we describe the initial formulation, as well as the procedure to separate the violated inequalities (3.3) and (3.4).

3.4.1 Initial relaxation

The algorithm starts with the integrality conditions relaxed and only a subset of constraints. For the MC-MDRPP, the initial subproblem includes constraints (3.2), (3.5) and (3.6), whereas for the MM-MDRPP the initial set of constraints consists of (3.2), (3.5), (3.6) and (3.10). A small subset of connectivity and parity constraints are considered. In particular, the initial relaxations are reinforced with the aggregate connectivity inequalities (3.13), (3.14) and (3.15), plus the aggregate parity constraints (3.17), associated with R-odd singletons, i.e., $S = \{v\}$ with $v \in V$ and $|\delta_R(v)|$ odd.

3.4.2 Separation of inequalities

Before presenting the separation procedures for connectivity and parity inequalities, we introduce some additional notation. Throughout $(\overline{x}, \overline{y})$ denotes the current LP solution and, for each depot $d \in D$, $(\overline{x}^d, \overline{y}^d)$ the partial LP solution associated with depot d, i.e. the components of $(\overline{x}, \overline{y})$ associated with d. Furthermore, $G^d_{\overline{x},\overline{y}} = (V^d, E_{\overline{x}^d,\overline{y}^d})$ denotes the support graph of the partial solution $(\overline{x}^d, \overline{y}^d)$ for depot $d \in D$, obtained from G by eliminating all edges in E with $\overline{x}^d_e = 0$ and all vertices that are not incident with any edge of $E_{\overline{x}^d,\overline{y}^d}$.

Separation of connectivity inequalities (3.3)
The separation problem for connectivity constraints associated with a given depot $d \in D$ can be solved similarly to the connectivity constrainrs of the RPP (see Chapter 1.4.1.3). Thus, for each depot $d \in D$ we first check if $G^d_{\overline{x},\overline{y}}$ is connected. If it is not, each connected component C with vertex set $V(C) \subseteq V^d \setminus \{d\}$ defines a violated connectivity constraint (3.3) for depot d. Otherwise when $G^d_{\overline{x},\overline{y}}$ is connected we build the tree of min-cuts T^d of $G^d_{\overline{x},\overline{y}}$ with capacities given by $\overline{x}^d_e + \overline{y}^d_e$. Then, using an adaptation of Belenguer and Benavent (1998), for each edge $e = \{u, v\}$ in $E_{\overline{x}^d,\overline{y}^d}$ with $u, v \in V \setminus \{d\}$, the minimum cut $\delta(S)$ such that $e \in \gamma(S)$ is easily obtained from the min-cut tree T^d. If the value of the min cut is smaller than $2\overline{x}^d_e$ then the inequalities (3.3) associated with S and d are violated by $(\overline{x}^d, \overline{y}^d)$. This separation method is exact and similar to the procedure used by other authors to separate connectivity constraints in other ARPs Ahr (2004), Aráoz et al. (2009a), Corberán et al. (2011).

Separation of parity inequalities (3.4)
The algorithm that we use follows the spirit of the procedures used by other authors with similar parity constraints for other ARPs with binary variables

Aráoz et al. (2009a), Aráoz et al. (2009b), Corberán et al. (2011). Now the separation is applied for each $d \in D$. The algorithm starts by building the tree of min-cuts of the support graph $G^d_{\overline{x},\overline{y}}$, T^b, with capacities vector b defined as $b_e = min\{(\overline{x}^d_e - \overline{y}^d_e), 1-(\overline{x}^d_e - \overline{y}^d_e)\}$. If the selected cut-set contains edge e, this criterion dictates whether edge e should be assigned to H or to $\delta(S)\setminus H$ in order to obtain the smallest possible value of the left hand side of (3.4). When T^b has a cut $\delta(S)$ of capacity smaller than one, i.e. $b(\delta(S)) < 1$, we consider its vertex set S, and the set of edges $H = \{e \in \delta(S) \mid (\overline{x}^d_e - \overline{y}^d_e) \geq 0.5\}$, which, as explained, produces the smallest possible value on the left hand side of (3.4). When $|H|$ is odd, H defines, together with S, a violated inequality of type (3.4). Otherwise, if $|H|$ is even, by moving one edge from H to $\delta(S) \setminus H$ or viceversa, so the new set H will be odd. Let Δ the increment derived from the edge movement computed as $\Delta = min\left\{\min\{\overline{x}^d_e - \overline{y}^d_e : e \in \delta(S) \setminus H\}, \min\{1 - (\overline{x}^d_e - \overline{y}^d_e) : e \in H\}\right\}$. When $b(\delta(S)) + \Delta < 1$, the updated set H defines a violated inequality (3.4) for d and S for the solution $(\overline{x}^d, \overline{y}^d)$.

It is possible that the minimum cut-set of T^b does not produce a violated inequality (3.4) even it exists. This could happen only if the set H associated with the minimum cut in T^b is even. Fortunately, in Letchford and Salazar-González (2015) it is proven that exploring all cut-sets of T^b as explained above defines an exact algorithm for knowing whether a violated inequality (3.4) exists. The order of such an algorithm is dominated by that of the algorithm that obtains the min-cut tree T^b. In practice, however, this upper limit on the order of the algorithm is very seldom reached. On the one hand, each connected component of $G^d_{\overline{x},\overline{y}}$ for the capacities vector b already defines some of the subsets S of the tree T^b and connected components can be obtained with a small computational burden. On the other hand, when $G^d_{\overline{x},\overline{y}}$ defines one single connected component but a violated inequality exists, most often the cut-set producing the violated inequality will be identified before completing the full cut-tree T^b. Thus, in most cases, only if no violated inequality (3.4) exists it will be necessary to compute all the min-cuts that define T^b.

3.5 Computational experience

In order to evaluate the performance of the branch-and-cut algorithms described above, we have run a series of computational experiments. Programs have been coded in C++ using CPLEX 12.5 Concert Technology for the solution of the LP relaxations. The maximum computing time has been set to four hours. Moreover, the cuts generated by CPLEX have been disabled. The experiments were run on an Intel Core 2 CPU, 2.67 GHz and 8.00 GB RAM.

3.5.1 Set of benchmark instances

Since there were no available MDRPP benchmark instances, we have generated test instances from 118 well-known RPP benchmark instances. The original RPP benchmark instances are divided in five groups. The first group ALB contains two data sets ALBAIDAA and ALBAIDAB, obtained from the Albaida, Spain Graph (see Corberán and Sanchis, 1994, 1998). The second group contains the 24 instances, labeled P, of Christofides et al. (1981). The last 3 groups contain instances from Hertz et al. (1999): 36 instances with vertices of degree 4 and disconnected required edges sets (labeled D), 36 grid instances (labeled G), and 20 randomly generated instances (labeled R). In all cases we inherited the set of required edges and the cost function c from the original RPP instances.

Concerning the set of depots, we have considered two different cases: two and four facilities. Depots have been chosen randomly from the set of vertices, fulfilling that no connected component has more than one facility. For this, for each selected number of depots $|D| \in \{2, 4\}$ we have proceeded as follows. First, we randomly generate $|D|$ different numbers, k_i, $i = 1, \ldots, |D|$, from an integer uniform distribution $U[1, |K|]$, which give the indices of the clusters were the depots are located. Then, for each selected cluster, k_i the index of the vertex of V_{k_i} that becomes the facility is 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 2 and 4 depots, the instances that have fewer than four connected components have been removed from the experiment. Finally, the experiments have been run with two groups of 103 instances each.

Table 3.1: Summary of the instances

	♯ inst	$\|V_0\|$	$\|E_0\|$	$\|R\|$	$\|K\|$	$\|V\|/\|V_0\|$	$\|E\|/\|E_0\|$
ALB	2	90-102	144-166	88-99	10-11	1.00	0.99
P	17	7-50	13-184	7-78	2-8	1.00	0.99
D16	6	16	31	3-16	2-5	0.83	0.80
D36	9	36	72	10-38	4-11	0.78	0.79
D64	9	64	128	27-75	5-15	0.82	0.83
D100	9	100	200	50-121	9-22	0.85	0.87
G16	7	16	24	3-13	3-5	0.74	0.67
G36	9	36	60	11-35	5-9	0.79	0.75
G64	9	64	112	24-68	4-14	0.80	0.78
G100	9	100	180	41-113	4-20	0.83	0.83
R20	2	20	47-75	3-7	3-4	0.48	0.36
R30	5	30	70-112	7-11	4-6	0.47	0.41
R40	5	40	82-203	8-18	5-9	0.50	0.50
R50	5	50	130-203	13-20	6-12	0.50	0.54

Table 3.1 depicts information on these instances, which have been grouped according to their characteristics and sizes. The meaning of the columns is as follows: column under ♯ *inst* gives the number of instances in the group; columns under $|V_0|$ 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 those required edges. In the above columns, when not all the instances of the group had 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, columns under $|V|/|V_0|$ and $|E|/|E_0|$ respectively correspond to the average ratios of the number of vertices or edges in the transformed graph related to the original graph. As it is known, the transformed graph is considerably smaller than the original graph, in terms of the number of vertices and edges.

3.5.2 Results for Min-Cost Multi-Depot RPP

The results for the MC-MDRPP for the instances with two and four depots are summarized in Tables 3.2 and 3.3, respectively. For each group of instances, columns 2-5 give information about the root node of the enumeration tree, while columns 6-11 give the results of the search tree. Column under $\sharp Opt_0$ shows the number of instances in the group that have been optimally solved in the root node. 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 two columns, under $CutsC_0$ and $CutsP_0$ give the average number of connectivity (3.3), and parity (3.4) cuts generated at the root node, respectively. Similarly, the next four columns under ♯*Opt*, *Gap*, *CutsC* and *CutsP* give the same information at termination: number of instances that have been optimally solved, the average percentage gap with respect to the optimal or best-known solution, and the average number of connectivity and parity cuts generated after the root node, respectively. Column under $Nodes$ shows the average number of nodes that were explored in the search tree. Finally, the column under *CPU (s)* gives the overall computing time in seconds. These times do not include the preprocessing time for the reduction of the graph neither the time for loading the formulation, which are negligible as compared to the solution times reported in the tables.

The results show that our algorithm found the optimal solution for almost all benchmark instances, with the exception of some instances belonging to the largest instance groups (D100 and G100). Particularly, for 36 2-depot instances, a provable optimal solution was obtained already at the root node. While at termination, optimality of the current solution was proven for 100 of the 103 2-depot instances. The unsolved instances are D35, G33, and G34 which, as mention, belong to groups D100 and G100. Respectively, their percentage optimality gap at termination are 6.34%, 9.88%, and 15.36%. Instead, for the 4-depot instances, optimality was proven at the root node for 53 instances and at termination for 95 of the 103 benchmark instances. Furthermore, no feasible integer solution was found within the time limit for any of the eight unsolved instances: two instances in group D100 (D34, D35) and six instances in group G100 (G30–G35).

Table 3.2: Summary of results for MC-MDRPP for two-depot instances

	$\sharp Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$\sharp Opt$	Gap	$CutsC$	$CutsP$	$Nodes$	$CPU(s)$
ALB	0/2	2.40	3568	153	2/2	0	6889.50	311	10	200.18
P	5/17	2.97	472.47	33.71	17/17	0	318.06	50.24	1.81	1.87
D16	6/6	0	56.83	8.33	-	-	-	-	0	0.03
D36	1/9	0.92	366.67	39.89	9/9	0	149	32.33	5.67	0.60
D64	0/9	1.59	1635.22	80.56	9/9	0	1516.33	178.67	20.22	16.24
D100	0/9	4.11	4392.78	135.56	8/9	0.70	26876.56	1483.89	376.67	2452.42
G16	5/7	1.52	23.14	12.29	7/7	0	12.71	7.14	1.43	0.03
G36	3/9	1.72	313.67	43.22	9/9	0	189.44	32.67	2.33	0.53
G64	2/9	1.75	1474.89	93.89	9/9	0	7733.11	662.11	164.11	156.77
G100	0/9	4.59	14368.89	422.44	7/9	2.80	59850.33	25381.78	337.44	4631.05
R20	2/2	0	7.50	5.50	-	-	-	-	-	0.02
R30	4/5	0.23	59.80	11.60	5/5	0	5.60	6.8	3	0.10
R40	4/5	0.09	330.60	23.60	5/5	0	100.40	18	3.4	0.28
R50	4/5	0.35	351.40	32.20	5/5	0	67.6	2.40	0.4	0.17

Table 3.3: Summary of results for MC-MDRPP for four-depots instances

	$\sharp Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$\sharp Opt$	Gap	$CutsC$	$CutsP$	$Nodes$	$CPU(s)$
ALB	0/2	1.8	18263	429	2/2	0	45754	1325	129	5476.70
P	11/17	0.73	455.94	49.88	17/17	0	2103.06	134.65	1.81	44.77
D16	6/6	0	0.50	1.33	-	-	-	-	-	0.01
D36	4/9	0.87	488.89	69.67	9/9	0	219.33	35.22	1.89	0.96
D64	1/9	2.27	4027	196.33	9/9	0	3509.78	1543.33	40.11	108.64
D100	0/9	24.60	17860.11	480.11	7/9	22.22	51504.33	1878.44	130	7085.23
G16	7/7	0	1.86	7.8	-	-	-	-	-	0.01
G36	5/9	1.56	340.78	69.78	9/9	0	795.22	98.22	45.44	10.50
G64	5/9	0.67	2636.67	196.89	9/9	0	16248.78	931.22	128	1835.31
G100	1/9	67.08	12813.33	548.22	3/9	66.67	56311.78	1341.67	38.33	9640.11
R20	2/2	0	1.5	7.5	-	-	-	-	-	0.02
R30	4/5	1.63	21.80	14.40	5/5	0	29.40	7.40	1.20	0.08
R40	4/5	0.13	352.60	42.20	5/5	0	20	4	1.80	0.35
R50	3/5	0.48	533.60	62.60	5/5	0	81.20	13.60	0.80	0.45

The computational effort required for solving the instances to optimality, can be evaluated by the required computing times. In this sense, only 5 instances with two depots and 14 of instances with four depots required more than one hour (including those instances for which no feasible solution was found within the time limit). Moreover, for 80% of two-depot and 74% of four-depot instances, respectively, the optimal solution was found in less than 1 minute. Moreover, comparing the difficulty to solve instances with two and four depots in terms of the required computing times, it can be seen that the algorithm is, in general, faster when the instances have fewer depots. Nevertheless, we can observe that the proposed algorithm was able to solve at the root node more four-depot than two-depot instances, even if the former involve a larger number of variables. A further analysis of those instances reveals a pattern, being instances with a reduced number of vertices and connected components.

We finally point out the small effect the assumption that service to required edges in the same connected can be split among different depots; in most solutions all connected components are fully served from the same depot. In particular, split components appear only in 22 of the two-depot instances and in 30 of the four-depot instances. This represents 21% and 29%, of the total number of consideres instances, respectively.

3.5.3 Results for Min-Max Multi-Depot RPP

A new series of computational experiments has been run for the MM-MDRPP. For these experiments we have considered the 78 two-depot instances with up to 64 nodes, and the 60 four-depot instances that consists of all instances with up to 50 nodes. We have not considered the sets of larger instances, since it was not possible to solve them to optimality within the time limit, and the percentatge optimality gaps at termination were big. Moreover, in most cases, even no feasible solution was known at termination. The results are summarized in Tables 3.4 and 3.5. The columns information is the same as before.

Table 3.4: Summary of results for MM-MDRPP for two-depot instances

	$\sharp Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$\sharp Opt$	Gap	$CutsC$	$CutsP$	$Nodes$	$CPU(s)$
P	1/12	8.31	319.92	27.75	12/12	0	678.58	156.42	35.58	4.43
D16	3/6	3.97	69.50	17.17	6/6	0	35.17	21.67	5.17	0.14
D36	0/9	12.59	448.78	41.78	9/9	0	1310.44	193.11	44.44	4.66
D64	0/9	11.21	2084.78	88.33	8/9	0.11	17340.33	1484.22	1688.44	1866.19
G16	4/7	9.71	38.43	13.86	7/7	0	18.43	13	3.71	0.04
G36	2/9	6.12	379.67	43.78	9/9	0	1279.56	224.56	27.78	5.25
G64	0/9	7.82	2720.33	122.44	9/9	0	25627.78	2532.00	624.56	1665.05
R20	1/2	13.11	18.00	5	2/2	0	55.50	23.50	7	0.16
R30	1/5	10.00	112.20	16.40	5/5	0	92.40	10.80	3.40	0.15
R40	2/5	3.47	418.20	23.40	5/5	0	650	109.20	59	2.42
R50	0/5	18.49	538.00	25.20	5/5	0	2749.40	197.60	55.80	1.41

The proposed algorithm found the optimal solution for all the tested MM-MDRPP instances but one, the exception being instance D26. The obtained results are consistent with those obtained for the MC-MDRPP. In general, 2-depot instances are easier to solve than 4-depot ones in terms of the size of the exploration tree and the computing time. Likewise, there are more 4-depot instances which are solved at the root node. This correspon also to small instances with a reduced number of vertices and connected components.

A comparison of the results of both models shows that, in general, the gap at the root node, the number of cuts, the number of explored nodes and the computing times are worse in the MM-MDRPP than in the MC-MDRPP. Another aspect that is relevant in the comparison between both models is the structure of the solutions. As could be expected, the overall routing costs

Table 3.5: Summary of results for MM-MDRPP for four-depot instances

	$\sharp Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$\sharp Opt$	Gap	$CutsC$	$CutsP$	$Nodes$	$CPU(s)$
P	2/12	14.24	661.83	69.58	12/12	0	4578.58	661.67	1107	1227.46
D16	4/6	5.60	38.83	21.50	6/6	0	0	1.33	1.17	0.08
D36	1/9	20.06	1153.56	122.33	9/9	0	319.56	490.78	176.56	47.03
G16	6/7	3.17	11.86	13.43	7/7	0	0	0	0.14	0.02
G36	2/9	15.90	805.78	124.78	9/9	0	3662.22	679.78	295.22	1117.09
R20	2/2	0	19.50	16	-	-	-	-	-	0.05
R30	3/5	6.73	144.20	30.20	5/5	0	82	24.20	15.50	0.38
R40	0/5	31.09	638.20	61.80	5/5	0	2370.40	306.40	120	30.25
R50	0/5	36,78	1005.60	92.80	5/5	0	5808.60	1152,60	815.2	113.81

are, in general, higher in optimal MM-MDRPP solutions than in optimal MC-MDRPP solutions. Even if there are 19 2-depot instances and 20 4-depot instances where the overall length of all routes is the same in both models, the average overall length increase is 13.09% for the 2-depot instances and 21.14% for the 4-depot instances. The maximum increases are 52.80% in instance R17 with two depots, and 73.69% in instance R11 with four depots. Nevertheless, we can also observe that optimal routes for MC-MDRPP tend to be unbalanced. In particular, when using model MM-MDRPP, the length of the maximum route usually decreases. On average the makespan decreases in 19% for the 2-depot instances with a maximum decrease of 46.15% in instance G25, and 27.20% for the 4-depot instances with a maximum decrease of 64.71% in instance G16. In fact, only in 13 of the 2-depot instances and in 21 of the 4-depot instances the length of the longest route remains the same on both models.

3.5.4 Balancing the length of the routes from one single depot

Finally, a last series of experiments has been run. In these experiments we have solved the undirected MM-K-RPP, which considers K vehicles located at a one single facility, and minimizes the length of the longest route. As mentioned in the introduction, the MM-K-RPP is a particular case of the MM-MDRPP, by considering K depots co-located at the same vertex and performing one single route from each co-located depot. For the experiments, all instances with up to 40 nodes have been considered, one of the facilities has been randomly selected and replicated K times while all other previous depots have been ignored.

Tables 3.6 and 3.7 summarize the results obtained for $K \in \{2, 4\}$. Comparing optimal solutions to the MM-K-RPP and the MM-MDRPP, we observe that the cost of the longest route in optimal MM-K-RPP solutions increases considerably in comparison to that of the MM-MDRPP. Consequently, the total routing cost increases as well. The average makespan increases in 26.84% for the two-depot instances and in 102% for the four-depot instances. This represents a total increase of the overall length of 31.29 % and 116.50% respectively, on average.

Table 3.6: Summary of results for MM-K-RPP for two-routes instances

	$\sharp Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$\sharp Opt$	Gap	$CutsC$	$CutsP$	$Nodes$	$CPU(s)$
P	0/12	24.29	788.33	39.58	12/12	0	2641.25	369.33	286.00	25.43
D16	0/6	30.80	125.00	17.67	6/6	0	317.50	117.83	52.17	0.55
D36	0/9	21.49	639.78	40.44	9/9	0	5104.67	703.89	858.00	1685.29
G16	1/7	30.43	67.86	18.29	7/7	0	135.43	37.86	27.14	0.23
G36	0/9	24.89	692.11	49.56	9/9	0	4032.22	556.00	252.67	368.37
R20	0/2	33.48	47.50	15.00	2/2	0	133.50	34.00	18.00	0.31
R30	0/5	22.35	147.60	21.40	5/5	0	473.80	77.40	57.80	0.88
R40	0/5	14.74	432.00	30.00	5/5	0	4079.40	441.60	414.80	28.81

Concerning the operational aspect, the computational effort required for solving the MM-K-RPP instances to optimality is higher than in the previous experiments, for instances of the same size and characteristics. In comparison with the MM-MDRPP, the computing time of the MM-K-RPP increases around 1604% for two-depot instances and 644% for four-depot instances. Furthermore, the number of instances that could not be optimally solved within the time limit increased to eight for the four-depots instances. We attribute this increase in the difficulty for optimally solving the instances to the symmetry that now appears for the routes, as they can now be interchanged.

Table 3.7: Summary of results for MM-K-RPP for four-routes instances

	$\sharp Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$\sharp Opt$	Gap	$CutsC$	$CutsP$	$Nodes$	$CPU(s)$
P	0/12	37.57	1812.50	93.58	10/12	12.22	14981.33	1822.83	3422.25	3682.92
D16	0/6	48.76	236.17	42.50	6/6	0	1013.67	446.83	594.83	15.40
D36	0/9	36.27	2132.33	167.78	7/9	7.10	20157.67	2088.78	11910.89	3871.17
G16	1/7	32.54	153.86	57.57	7/7	0	301	112.86	72.71	2.10
G36	0/9	47.21	2974.44	204	8/9	3.54	14838.67	1694	1127	4502.25
R20	0/2	50.43	257.50	31.50	2/2	0	812.50	625	692.50	20.09
R30	0/5	47.38	448.80	47.40	5/5	0	2249.80	478.80	442.20	44.59
R40	0/5	40.99	1090	75.80	2/5	6.60	19397.40	2650	4209.40	3336.25

Chapter 4

Aggregate formulation for the Min-Cost Multi-Depot RPP

In this chapter we propose a compact integer linear programming formulation for the MC-MDRPP. This formulation is based on an aggregate view of the decision variables, which now only have two indices. Furthermore, a polyhedral analysis of the presented formulation has been developed, as well as a branch-and-cut algorithm, which has been tested on a large set of benchmark instances, involving up to 744 vertices, 140 required components and 1000 required edges. Most of the content of this chapter has been published in Fernández et al. (2018).

4.1 Aggregate decision variables

As an alternative to the natural modeling option where variables are associated with edges and facilities, the aggregate view yields to only two-index variables, which are solely associated with edges, but not with facilities. This modeling option reduces the number of decision variables. However, the decrease in the amount of decision variables comes at the expense of additional difficulties.

Figure 5.2, where gray squares represent depots and solid lines the required edges, illustrates that, with the new view, connectivity and parity constraints are not sufficient to guarantee well-defined routes. Observe that the displayed solution is not feasible despite being connected and of even degree at all nodes, as it is not possible to decompose the solution into three routes, each of them starting and ending at the same facility. In the formulation that we propose, the difficulty for stating well-defined routes is overcome through a new set of constraints, which can be separated in polynomial time and that we present afterwards.

Taking into account the optimality condition O5 presented in Chapter 3.1.2, in an optimal MC-MDRPP solution an edge can be traversed multiple times. In

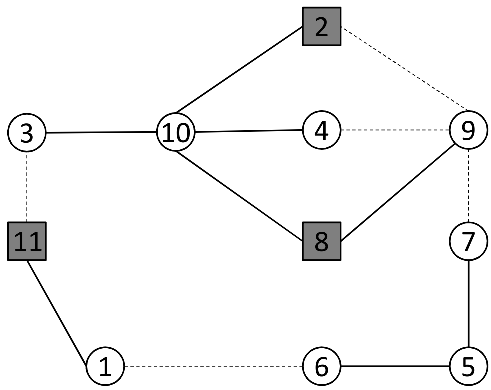


Figure 4.1: Infeasible solution satisfying connectivity and parity constraints

particular, required edges plus edges from the MST, T_C can be traversed $2|D|$ times, whereas the remaining edges can be traversed $|D|$ times. Thus, unlike the three-index formulation, the aggregate formulation can not use two sets of binary variables unless we operate on a complete graph. As it was pointed out at the end of Chapter 3.1.2 when $G = (V, E)$ is a complete graph, optimality conditions O6 and O7 apply. Therefore, there exists an optimal solution to MC-MDRPP in which no edge is traversed more than twice. Moreover, the only edges that can be traversed twice are required edges connecting two R-odd vertices plus edges satisfying O3.

In the reminder of this chapter, unless otherwise stagted, we assume that $G(V, E)$ is a complete graph.

4.2 Two-index formulation

Next we propose a formulation for the MC-MDRPP stated on a complete graph, which uses two-index binary variables only. In particular, two sets of binary variables are used, associated with the first and second traversal of edges. For each $e \in E$, let x_e be a binary variable indicating whether or not edge e is traversed by some route. We denote by $E^y \subset E$ the set of edges that can be traversed twice in an optimal solution, which, according to the optimality condition O7, consists of the required edges connecting two R-odd vertices plus the edges of T_C. For each $e \in E^y$, let y_e be a binary variable that takes the value one if and only if edge e is traversed twice.

Then, an integer linear program for the MC-MDRRP is as follows:

$$\text{minimize} \sum_{e\in E} c_e x_e + \sum_{e\in E^y} c_e y_e \tag{4.1}$$

subject to

$$
\begin{array}{lll}
(x+y)(\delta(d)) \geq 2 & d \in D & (4.2)\\
(x+y)(\delta(S)) \geq 2 & S \subseteq V \setminus D & (4.3)\\
(x-y)(\delta(S)\setminus H) + y(H) \geq x(H) - |H| + 1 & S \subset V,\ H \subseteq \delta(S) & (4.4)\\
 & |H| \text{ odd} & \\
(x-y)(Q) + y(H) \geq x(H) - |H| + |D'| & S \subset V \setminus D, & (4.5)\\
 & D' = \{d_i\}_{i\in I} \subset D,\ |D'| > 1, & \\
 & H_i \subseteq \delta(S) \cap \delta(d_i),\ |H_i| \text{ odd}, & \\
 & H = \bigcup_{i\in I} H_i, & \\
 & Q = (\delta(S)\setminus H) \setminus \big(\delta(D\setminus D')\big) & \\
x_e = 1 & e \in R & (4.6)\\
y_e \leq x_e & e \in E^y & (4.7)\\
x_e \in \{0,1\} & e \in E & (4.8)\\
y_e \in \{0,1\} & e \in E^y. & (4.9)
\end{array}
$$

Inequalities (4.2) ensure that all depots are used. Constraints (4.3) are the well-known connectivity inequalities. They impose that at least two edges cross the cut-set $\delta(S)$ of any set of vertices S that does not contain a facility, i.e. $S \subset V\backslash D$. Inequalities (4.4) ensure the parity (even degree) of the solution exploiting the precedence relationship of the x variables with respect to the y variables imposed by constraints (4.7). Inequalities (4.5) are new and will be referred to as *Return-to-facility constraints (Rt-FCs)*. 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 4.2.1, 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 (4.6) ensure that each required edge is served in the solution, whereas inequalities (4.7) 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 (4.8) and (4.9).

The above formulation contains $|E|$ x variables and $|E^y|$ y variables. There are $|D|$ constraints of type (4.2), $|R|$ equalities (4.6), and $|E^y|$ constraints of type (4.7). The size of the families constraints (4.3), (4.4), and (4.5) is exponential in $|V|$.

Proposition 4.2.1. *Formulation* (4.2)–(4.9) *is valid for the MC-MDRRP.*

Proof. We will show that if a solution $(\overline{x}, \overline{y})$ satisfying (4.2)–(4.4), (4.6)–(4.9) is not feasible for the MC-MDRRP, then there exists a constraint (4.5) 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 (4.3)–(4.4), if $(\overline{x}, \overline{y})$ is not feasible for the MC-MDRRP then there must be a simple tour T traversing at least two facilities. 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 Rt-FCs (4.5) 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 5.2 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 Rt-FCs inequality (4.5) 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 4.2.1. An additional consequence of the above proof is that the Rt-FCs (4.5) associated with subsets D' with two facilities are enough to guarantee that the proposed formulation is valid.

4.2.1 Polyhedral analysis

Considering that all required edges must be traversed at least once, the MC-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 facility. Accordingly, we can reformulate (4.2)–(4.9) by slightly modifying the meaning and the domain of the variables. Now 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 Chapter 3.1.2, we redefine the domain of the x variables as $E_x^2 \subset E$, 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 $E_y^2 \subset E$ 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 MC-MDRPP can be expressed as:

$$\sum_{e \in R} c_e + \text{minimize} \left\{ \sum_{e \in E_x^2} c_e x_e + \sum_{e \in E_y^2} c_e y_e \right\} \tag{4.10}$$

subject to

$$
\begin{array}{lll}
x(\delta(d)) \geq 2 & d \in D & (4.11) \\
(x+y)(\delta(S)) \geq 2 & S = \cup_{i \in K'} V_i \setminus D & (4.3) \\
 & \emptyset \neq K' \subset K & \\
(x-y)(\delta(S) \setminus H) + y(H) \geq x(H) - |H| + 1 & S \subset V,\ R\text{-even} & (4.4) \\
 & H \subseteq \delta(S),\ |H| \text{ odd} & \\
(x-y)(\delta(S)) \geq 1 & S \subset V,\ R\text{-odd} & (4.12) \\
(x-y)(Q) + y(H) \geq x(H) - |H| + |D'| & S \subset V \setminus D, & (4.5) \\
 & D' = \{d_i\}_{i \in I} \subset D,\ |D'| > 1, & \\
 & H_i \subseteq \delta(S) \cap \delta(d_i),\ |H_i| \text{ odd}, & \\
 & H = \bigcup_{i \in I} H_i, & \\
 & Q = (\delta(S) \setminus H) \setminus \big(\delta(D \setminus D')\big) & \\
y_e \leq x_e & e \in E_y^2 & (4.7) \\
x_e \in \{0,1\} & e \in E_x^2 & (4.8) \\
y_e \in \{0,1\} & e \in E_y^2. & (4.9)
\end{array}
$$

Next we study the polyhedral properties of (4.10)–(4.9). We denote by $P_{(MC-MDRPP)}$ the polytope defined by the convex hull of feasible solutions to the above formulation: $P_{(MC-MDRPP)} = conv\{(x,y) \in \{0,1\}^{|E_x^2|+|E_y^2|} : (x,y)$ satisfies (4.11)–(4.7)}.

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_x^2, they correspond to T-joins, connecting given pairs of vertices, that only use *true* edges of the set E_x^2. 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_x^2. 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 4.2.2. *$P_{(MC-MDRPP)}$ 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 Ghiani and Laporte (2000) 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, $P(MC - MDRPP) \subset \{x : x_e = 1\}$. 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 $P(MC - MDRPP) \subset \{x : x_{e^{(1)}} = 1$ and $x_{e^{(2)}} = 1\}$. Otherwise, if $\delta(S)$ is R-even, $P(MC - MDRPP) \subset \{x : x_{e^{(1)}} = x_{e^{(2)}}\}$, and if $\delta(S)$ is R-odd, $P(MC - MDRPP) \subset \{x : x_{e^{(1)}} + x_{e^{(2)}} = 1\}$. 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 $P(MC - MDRPP) \subset \{x : x_{e^{(1)}} + x_{e^{(2)}} + x_{e^{(3)}} = 2\}$.

The condition is sufficient. Let us find $|E_x^2| + |E_y^2| + 1$ affinely independent solutions satisfying the connectivity, parity and return-to-facility constraints.

The first solution, denoted by (x^0, y^0), contains one traversal of the edge connecting each R-odd vertex with an arbitrarily chosen facility $d_0 \in D$, plus two traversals of all the edges of T_C. To guarantee the parity of the depots in the solution, it may be necessary to add some edges connecting some pairs of facilities. 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.

- $a1)$ 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 $f \in E_y$.
- $a2)$ 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 $f \in E_y$.
- $a3)$ 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., $x_f^e = x_f^0$, for all $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y^2$.

c) Case $e \in E_x^2 \cap E_y^2$. Now $x_e^0 = y_e^0 = 1$. We now generate two solutions: (x^e, y^e), associated with $e \in E_x^2$, and (x'^e, y'^e), associated with $e \in E_y^2$. For (x^e, y^e) we keep $x_e^e = 1$ but set $y_e^e = 0$. Then we set $x_{e_u}^e = x_{e_v}^e = 1$ 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 facility). 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 $P_{(MC-MDRPP)}$. The proofs of these results follow a similar spirit of those for the RPP in Ghiani and Laporte (2000).

Proposition 4.2.3. *The inequality $x_e \geq 0$ defines a facet of $P_{(MC-MDRPP)}$ 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 Ghiani and Laporte (2000) directly applies to the MC-MDRPP, independently of the number of depots. The face $\{x \in P_{(MC-MDRPP)} : x_e = 0\}$ has the same dimension as the polytope associated with the MC-MDRPP defined on the graph obtained after removing edge e from G. ■

Proposition 4.2.4. *The inequality $x_e \leq 1$ induces a facet of $P_{(MC-MDRPP)}$ 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 $|E_x^2| + |E_y^2|$ 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 4.2.2. Recall that $x_e^e = 1$. The remaining $|E_x^2| + |E_y^2| - 1$ solutions may be constructed following the same process as in Proposition 4.2.2, modifying in each new solution one of the other components. ■

Proposition 4.2.5. *The connectivity inequality* (4.3) *associated with* $S = \bigcup_{i \in K'} V_i$ ($\emptyset \neq K' \subset K$)*,* $S \bigcap D = \emptyset$*, induces a facet of* $P_{(MC-MDRPP)}$ *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 (4.3) associated with $G(S)$ is dominated by the connectivity inequality (4.3) corresponding to $G(S_1)$. A similar situation arises if some component of $G(V \backslash S)$ contains no facility. 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 (4.3) 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 4.2.2.

- $a1)$ 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 $f \in E_y^2$.
- $a2)$ 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 $f \in E_x^2 \setminus \{e\}$, $y_f^e = y_f^0$, for all $f \in E_y$.

$a3)$ Case u R-odd and v R-even (or vice versa). In this case $x^0_e = 0$ 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. $x^0_{e_u} = 1$ and $x^0_{e_v} = 0$. We set $x^e_e = 1 - x^0_e = 1$, $x^e_{e_u} = 1 - x^0_{e_u} = 0$, and $x^e_{e_v} = 1 - x^0_{e_v} = 1$, so 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 $f \in E_y$.

$b)$ Case $e = \{u, v\} \in E^2_x \setminus E^2_y$ with u, v in different components. In this case $x^0_e = 0$, 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 $x^0_{e'} = y^0_{e'} = 1$. 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 4.2.6. *The parity inequalities* (4.4) *and* (4.12) *induce facets of* $P_{(MC-MDRPP)}$ *if and only if the following conditions hold: i) for every subset* $S' \subset S$ *(or* S' *in* $V \setminus S$*) with* $S' \cap D = \emptyset$*, then* $|\delta(S') \setminus \delta(S)| \geq 2$*. 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 Ghiani and Laporte (2000), based on Barahona and Grötschel (1986), directly applies to the MC-MDRPP, independently of the number of depots. ■

Proposition 4.2.7. *The Rt-FCs* (4.5) *induce facets of* $P_{(MC-MDRPP)}$ *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 Rt-FCs are an adaptation of the parity inequalities. So, as in the previous case, the proof in Ghiani and Laporte (2000), based on Barahona and Grötschel (1986), directly applies to the Rt-FCs. ■

Note that in Proposition 4.2.7 condition $ii)$ of Proposition 4.2.6 is no longer needed. The reason is that inequalities (4.5) are defined for $r > 1$. Therefore, they are never not dominated by the connectivity inequality (4.3) associated with S, as it happens for inequalities (4.4) when condition $ii)$ does not hold.

4.2.2 Dropping the completeness assumption for the input graph

As mentioned, the two-index formulation (4.2)–(4.9) operates on a complete graph. Thus, the memory requirements of any solution algorithm based on that formulatio, will become too high when the instance size increases. Unfortunately, when dealing with uncomplete graphs optimality condition O6 no longer holds. Therefore, MC-MDRRP cannot be modeled with binary variables x and y, as 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 (4.2)–(4.9) to the case of a generral undirected connected graph, 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:

$$\text{minimize} \sum_{e \in E} c_e x_e + \sum_{e \in E^y} c_e y_e \tag{4.13}$$

subject to

$$(x+y)(\delta(d)) \geq 2 \qquad d \in D \tag{4.14}$$

$$(x+y)(\delta(S)) \geq 2 \qquad S \subseteq V \setminus D \tag{4.15}$$

$$(x-y)(\delta(S) \setminus H) + y(H) \geq x(H) - |H| + 1 \qquad S \subset V, H \subseteq \delta(S), |H| \text{ odd} \tag{4.16}$$

$$(x-y)(Q) + (x+y)(Q') \geq (x-y)(H) - |H| + |D'| \qquad S \subset V \setminus D,\ D' = \{d_i\}_{i \in I} \subset D,\ |D'| > 1, \tag{4.17}$$

$$
\begin{aligned}
& && H_i \subseteq \delta(S) \cap \delta(d_i), \\
& && |H_i| \text{ odd}, \ H = \bigcup_{i \in I} H_i, \\
& && Q = (\delta(S) \backslash H) \cap \delta(D'), \\
& && Q' = (\delta(S) \backslash H) \backslash \delta(D) \\
& x_e = 1 && e \in R && (4.18) \\
& y_e \leq 2|D| x_e && e \in E && (4.19) \\
& x_e \in \{0, 1\} && e \in E && (4.20) \\
& y_e \in \mathbb{Z}_+ && e \in E. && (4.21)
\end{aligned}
$$

Note that the polyhedral analysis of Chapter 4.2.1 for the formulation for the complete graph, can be adapted to the general formulation since all solutions used in the proofs, where some edge e are traverseed only one additional time, $y_e = 1$, are still feasible for the general formulation.

4.3 Branch-and-cut algorithm

In this section, we present an exact branch-and-cut algorithm for the MC-MDRPP, based on the new aggregate formulation (4.2)–(4.9). As usual, the families of constraints of exponential size, as connectivity (4.3), parity (4.4), and Rt-FCs (4.5), are initially relaxed.

4.3.1 Initial relaxation

The algorithm starts with all integrality conditions relaxed and a subset of constraints. The initial formulation includes constraints (4.2), (4.6), and (4.7), plus a reduced subset of connectivity (4.3) and parity (4.4) constraints. In particular, we consider two subfamilies of the connectivity constraints (4.3) and a subset of the parity constraints (4.4). For connectivity inequalities, on the one hand, we include the inequalities associated with the subsets defined by the end-vertices of the edges not incident with any depot, i.e., $S = \{u, v \in V \setminus D\}$. On the other hand, we consider the inequalities associated with the subsets defined by the vertices of each component without any depot, i.e. $S = V_k$, $k \in K$, with $V_k \cap D = \emptyset$. As for the parity constraints, initially, we only incorporate the ones associated with R-odd singletons i.e., $S = \{v\}$ with $v \in V$ and $|\delta_R(v)|$ odd.

4.3.2 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 (4.5) are only separated when the LP solution $(\overline{x}, \overline{y})$ is integer, while inequalities (4.3) and (4.4) 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 (4.3), (4.4) and (4.5).

$a1)$ Connectivity inequalities (4.3). 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 (4.4). 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.

$a3)$ Return-to-facility inequalities (4.5). Violated inequalities can be easily identified by first finding a tour decomposition of the solution (applying, for instance, Hierholzer's algorithm Hierholzer, 1873) and then checking if any of the tours contains a path $P_{d_1 d_2}$ connecting two (consecutive) facilities. In this case $D' = \{d_1, d_2\}$ and $S = V(P_{d_1 d_2}) \setminus D'$ defines a violated cut.

$b)$ *Case* $(\overline{x}, \overline{y})$ *is fractional:* Check for violated inequalities of types (4.3) and (4.4).

In each case, we first apply a heuristic and only resort to the 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.

Separation of connectivity inequalities (4.3)
The separation for inequalities (4.3) 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 4.2 shows, violated connectivity constraints (4.3) 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 (4.3) 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 cut-sets 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 $(\overline{x} + \overline{y})(\delta(S)) = v^{V \setminus D}(S) + (\overline{x} + \overline{y})\,(\delta(S) \cap \delta(D))$.

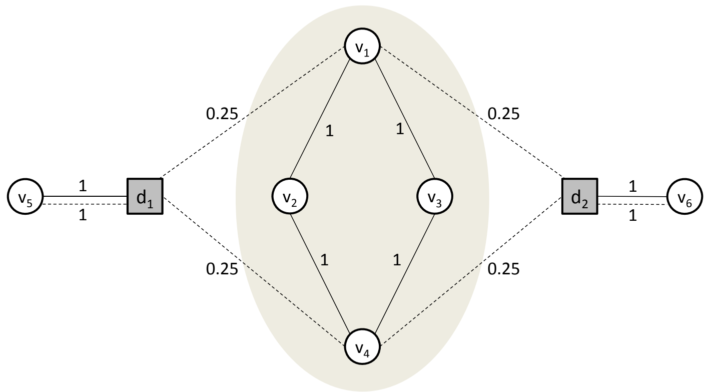


Figure 4.2: Violated connectivity constraint not associated with a min-cut

Heuristic separation: As usual, the heuristic for the separation of (4.3) looks for connected components, but now, 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 (4.3) associated with $V(C)$ is violated by $(\overline{x},\overline{y})$.

Exact separation: The exact separation of connectivity constraints (4.3) involves building the tree of min-cuts T of $G^{V\setminus D}(\overline{x},\overline{y})$ 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 (4.3) associated with S is violated by $(\overline{x},\overline{y})$.

Separation of parity inequalities (4.4)

The procedure for solving the separation problem of parity inequalities (4.4) takes the same scheme as in the separation of (3.4).

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 (4.4) associated with S and H is violated by $(\overline{x},\overline{y})$. Otherwise, if $b(\delta(S)) + \Delta < 1$, the parity constraint (4.4) 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 (4.4) 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* ; $i \leftarrow 1$

while $v^{S^i} < 1$ **and** end = *false* **do**

Define $H^i \subset \delta(S^i)$.
if($|H^i|$ odd **then**)
end ← *true* (constraint (4.4) 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 (4.4) violated by $(\overline{x}, \overline{y})$ for S^i and updated set H^i)
else
if $(i = r)$ **then**
end ← *true* (no violated constraint (4.4) exists)
else
$i \leftarrow i + 1$
end-if
end-if
end-if

Summarizing, the exact separation for inequalities (4.4) 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 as $b_e = \min\{(\overline{x}_e - \overline{y}_e), 1 - (\overline{x}_e - \overline{y}_e)\}$. It is important to recall that the smallest value of the left-hand side of inequality (4.4) after making H odd is not necessarily associated with the smallest min-cut of the tree.

4.4 Computational experience

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 4 or 24 hours depending on the instance size, 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 (4.4). The values for the threshold ε used in the heuristic separation of connectivity (4.3) and parity (4.4) constraints 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.

4.4.1 Set of larger benchmark instances

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, which contains 118 instances with up to 100 vertices, was already used in Chapter 3. The new set of benchmark instances contains larger instances with up to 750 vertices, and have been adapted from the "ALBA", "GRP" and "MADR" General Routing Problem instances and from the "URP" Undirected RPP instances Corberán et al. (2007).

The adaptation to the MC-MDRPP of the new set of instances preserves the set of required edges and well as the edges costs. Like with the set of smaller benchmark instances, two and four facilities have been arbitrarily chosen as the depots of each new two- or four-depot instance. Table 4.1 provides information on the new set of instances, which are grouped according to their characteristics and sizes. All the adapted instances are available at http://www.eio.upc.edu/en/homepages/elena/mdarp-instances.

Table 4.1: Summary of the instances

	# inst	$\lvert V_0\rvert$	$\lvert E_0\rvert$	$\lvert R\rvert$	$\lvert K\rvert$	$\lvert V\rvert/\lvert V_0\rvert$	$\lvert E\rvert/\lvert E_0\rvert$
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

4.4.2 Results for Min-cost Multi-Depot RPP

The set of small and medium instances with two and four depots was solved with formulation (4.1)–(4.9). For this, the original input graph was first completed by adding edges representing shortest paths connecting pairs of vertices not directly connected in the original graph. The results for the instances with two and four depots are summarized in Tables 4.2 and 4.3, 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. As in Chapter 3, the column under $\#Opt_0$ shows the number of instances 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

Table 4.2: Computational results for two-depot instances

	$\#Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$CutsD_0$	$\#Opt$	Gap	$CutsC$	$CutsP$	$CutsD$	$Nodes$	$CPU(s)$
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

known solution at termination. The following three columns, under the headings *CutsC*, *CutsP*, and *CutsD*, give the average number of connectivity (4.3), parity (4.4), and Rt-FCs (4.5) cuts generated, respectively. Similarly, the next five columns give the same information at termination. Column under $Nodes$ shows the average number of nodes explored in the search tree. Finally, the column under *CPU (s)* gives the total computing time in seconds.

As can be seen, the optimality of the current solution was proven for all the instances, independently of the number of facilities. Optimality was proven at

Table 4.3: Computational results for four-depot instances

	$\#Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$CutsD_0$	$\#Opt$	Gap	$CutsC$	$CutsP$	$CutsD$	$Nodes$	$CPU(s)$
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

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 exceptions were one ALB instance, which required 70 seconds, and two D100 instances, which required around 400 seconds each one. The three four-depot

D100 instances that exceeded one minute of computing time were optimally solved within less than two minutes.

Similarly to the algorithm for the 3-index formulation, the algorithm is, in general, faster for the two-depot instances than for the four-depot instances. Nevertheless, for instances with few vertices and 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 Rt-FCs (4.5) added 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.

The good results obtained for small and medium size instances, encouraged us to solve larger instances. However, the amount of memory required to complete the underlying graphs of these instances so as to solve them with formulation (4.1)–(4.9) becomes unaffordable. Hence, the new set of larger instances was solved with the general integer formulation (4.13)–(4.21). To this end, for the new benchmark sets ALB2, GRP, MAD; URP5 and URP7, after completing the input graph, we removed 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.

Tables 4.4 and 4.5 summarize the results obtained with the set of larger two- and four-depot instances with 116 to 744 vertices.

As can be seen, all 44 instances with up to 500 vertices (ALB2, GRP, MAD, and URP5) were 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 to solve these groups of instances. On the one hand, most of the instances with up to 200 vertices (ALB2, GRP and MAD), were solved within less than two minutes. Only three MAD instances required up to five minutes of computing time, whereas the most time consuming instance required nearly 30 minutes. On the other hand, the instances of group URP5 required several hours to be solved. The average computing time to solve URP5 instances was 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 could be solved within less than one and two hours, respectively. The maximum computing time for an instance of this group was around 13 hours. When four-depot instances are considered, only one instance was solved in less than two hours, and three of them required more than 12 hours. The maximum computing

Table 4.4: Computational results for big size two-depot instances

	$\#Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$CutsD_0$	$\#Opt$	Gap	$CutsC$	$CutsP$	$CutsD$	$Nodes$	$CPU(s)$
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 4.5: Computational results for big size four-depot instances

	$\#Opt_0$	Gap_0	$CutsC_0$	$CutsP_0$	$CutsD_0$	$\#Opt$	Gap	$CutsC$	$CutsP$	$CutsD$	$Nodes$	$CPU(s)$
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

time was 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 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 (computed 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.

4.4.3 Comparison of results: three-index vs two-index

Tables 4.6 and 4.7 provide comparisons between the results obtained with the disaggregate and aggregate formulations for instances with two and four depots, respectively. In terms of computing times the comparison is fair since all the experiments were performed on the same computer. Each table consists of two blocks of three columns each, the first one for a summary of results from the three-index formulation (3IF) and the second one for a summary of the results with the current branch-and-cut algorithm (referred to as 2IF). 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 (s)*).

Table 4.6: Comparison of tree- and two-index for two-depot instances

	3IF			2IF		
	$\#Opt_0$	$\#Opt$	$CPU(s)$	$\#Opt_0$	$\#Opt$	$CPU(s)$
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

As can be seen, the 2IF results notably outperform those of the 3IF. As mentioned, all instances are now solved in less than 400 seconds, whereas in the disaggregate formulation only 95% of the instances were solved to optimality within the time limit of 14,400 seconds. Furthermore, the number of instances optimally solved at the root node is notably larger than that of the three-index formulation, increasing from 36 to 60 for two-depot instances, and from 53 to 62 for four-depot instances. The results also show a significant reduction in the computing times of the two-index formulation with respect to the three-index one. On average, computing time decreases from 638.39 and 1746.08 seconds to 10.99 and 7.90 seconds, for the two and four depots instances, respectively.

Table 4.7: Comparison of tree- and two-index for four-depot instances

	3IF			2IF		
	$\#Opt_0$	$\#Opt$	$CPU(s)$	$\#Opt_0$	$\#Opt$	$CPU(s)$
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

Chapter 5

Location Arc Routing Problems

In this chapter we model and solve exactly several families of LARPs, which extend the MDRPPs to the case where the depots are not fixed in advance. We develop models that differ from each other in their objective function, whether there is an upper bound on the number of facilities to be located, or whether the facilities are capacitated. In particular, we consider two types of objective functions: min-cost objectives aiming at minimizing the overall routing costs, and min-max objectives aiming at minimizing the makespan. While some of the models assume that there are no capacity limitations, we also study problems that include a cardinality constraint on the number of users that can be served from an open facility. Finally some of the models ignore facilities set-up cost but include a limitation on the maximum number of facilities to be located, whereas in other models the number of open facilities is not limited but the facilities set-up are included in the objective function.

Two alternative formulations are presented, which use binary variables only. The first class uses disaggregated decision variables (three-index variables) that link routes with open facilities. All models can be handled with this type of formulation. The second class of formulations aggregates the information of all the routes. This leads to two-index variables, associated with the edges traversed by the routes, but that do not explicitly link them to the depots from which the routes operate. Like in the case of MDRPPs, this ap-

Table 5.1: Summary of models

	Objective function	Capacity	Limit on the number of open facilities
MC-p-LARP	Min routing cost	No	Yes
MM-p-LARP	Min makespan	No	Yes
MC-LARP	Min facilities set-up cost plus routing cost	No	No
MC-p-LARP-UD	Min routing cost	Yes	Yes
MM-p-LARP-UD	Min makespan	Yes	Yes
MC-LARP-UD	Min facilities set-up cost plus routing cost	Yes	No

proach reduces the number of required variables at the expense of presenting some additional difficulties. On one hand, only models minimizing cost without capacity constraint can be handled with this modeling technique. On other hand, a new set of constraints guaranteeing that the routes are consistent and return to their original depot is needed.

Both types of formulations are solved with branch-and-cut algorithms that demostrate a good performance through the obtained results, where instances with up to 200 vertices have been solved to optimality.

5.1 Formal definition

We consider LARPs defined on an undirected connected graph $G = (V, E)$, where V is the vertex set, and E is the edge set. Now, the set $D \subset V$ denotes a set of potential locations where facilities may be established. Like in the previous chapters $R \subset E$ denotes the set of required edges and the required components are denoted by $C_k = (V_k, R_k)$, $k \in K$. In addition to the traverasl cost associated with each edge $e \in E$, there is now a value $f_d \geq 0$, associated with each potential location $d \in D$, which indicates the set-up cost of *opening* a facility at d. Let p be an upper bound on the number of depots to be located. When there is a limitation on the service capacity of open facilities, we use b_d to denote the maximum number of required edges that can be served from a depot located at $d \in D$.

Feasible LARPs solutions consist of a subset of open facilities $D^* \subseteq D$, together with a set of non-empty routes, at least one for each selected facility, that serve all the required edges. Alternative objective functions or additional constraints characterize the different problems under study:

Definition 5.1.1.

- *The MC-p-LARP is to determine a feasible solution with at most p open facilites, i.e. $|D^*| \leq p$, that minimizes the sum of the routing costs.*

- *The MM-p-LARP is to determine feasible solution with at most p open facilites, i.e. $|D^*| \leq p$, that minimizes the makespan.*

- *The MC-LARP is to determine a feasible solution that minimizes the sum of the set-up costs of the selected facilities, plus the routing costs.*

We also consider capacitated versions of each of the above defined problems, where we assume that each required edge has a unit demand, and for each potential facility there is a constraint on the maximum demand that it can serve if it is opened. Since we consider unit demands, these capacitated versions reduce to cardinality constraints on the maximum number of required edges served by each facility. We denote by MC-p-LARP-UD, MM-p-LARP-UD, and MC-LARP-UD the capacitated versions of MC-p-LARP, MM-p-LARP, and MC-LARP, respectively.

5.1.1 Modeling assumptions

In the remainder of this chapter we assume that all opened facilities will be *used*, in the sense that there will be at least one non-empty route at each open facility. Note that, except for the LARPs with facilities set-up costs (MC-LARP and MC-LARP-UD), it is necessary to explicitly impose this condition since otherwise, alternative optimal solutions could exist, where some facility is open but never used. As we will see below, this basic requirement also justifies the hypothesis that *at most* p facilities be used, instead of the usual condition that *exactly* p facilities be opened. Intuitively, one could think that, when only routing costs are considered, opening more facilities would necessarily lead to solutions with smaller routing costs, since required edges could be served from *closer* depots. However, imposing to open (and use) exactly p facilities, may lead to suboptimal routing decisions or may even force the activation of a route that does not serve any required edge and deteriorates the value of the objective function. In 3.2 it was proven that the optimal value of an MDRPP where all depots must be used can asymptotically be twice the optimal value of the RPP on the same input graph. Indeed, this result can be extended to the MC-p-LARP and one can find instances where, asymptotically, the optimal value of an instance with p open facilities is twice the optimal value of the same instance with just one open depot.

Also for the case of the MM-p-LARP, forcing exactly p facilities to be opened may produce undesirable solutions. A simple example is given in Figure 5.1 which depicts two components and three potential locations for the facilities, where the solid lines represent required edges and the dotted lines the remaining edges. As can be seen, the optimal solution for the MM-p-LARP in that instance, when exactly two facilities must be opened, will activate facilities $L1$ and $L2$ and serve from each of them the required edges in their respective components. The makespan of that solution is three. This solution has a better objective than a solution in which three facilities are opened. Indeed when $p = 3$, facility $L3$ must also be opened and a route must be associated with it, for instance $(L3, B, L3)$, which does not serve any required edge, and gives an objective function value of four units.

Hence, we avoid potential awkward situations, like the one of the above example, by assuming that p represents the maximum number of facilities that can be opened, so the models that we study also dictate the optimal decision in terms of the number of facilities to open.

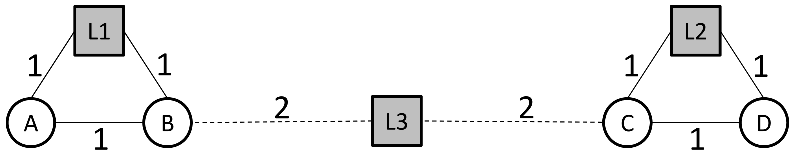


Figure 5.1: Example with better solution for $p = 2$ than for $p = 3$

Without loss of generality we also assume that $|D| \geq 3$. Indeed, if $|D| = 1$ no location decision must be made, so we just have an arc-routing problem. If $|D| = 2$ we can define an additional potential location placed at a fictitious node and connect it with only one vertex of V_R with an edge of cost greater than twice the sum of the costs of all other edges. This hypothesis will be used in the proofs of our polyhedral analysis, where we sometimes use three different depots to obtain the number of points that are needed.

5.1.2 Complexity and optimality conditions

The MC-MDRPP where the location of the facilities is known in advance, is a particular case of both the MC-p-LARP and the MC-LARP. Moreover, the MC-MDRPP is also a particular case of the MC-p-LARP-UD and the MC-LARP-UD, where the location of the facilities are known and there are no facilities capacity constraints. Similarly, the MM-MDRP is a particular case of both the MM-p-LARP and the MM-p-LARP-UD. Since the MC-MDRPP and the MM-MDRP are known to be NP-hard (see, 3.1.2), we have the following results:

Proposition 5.1.2.

- *The MC-p-LARP and the MC-p-LARP-UD are NP-hard*
- *The MM-p-LARP and the MM-p-LARP-UD are NP-hard.*
- *The MC-LARP and the MC-LARP-UD are NP-hard.*

All the formulations that we propose for the LARPs that we study use only binary variables. This follows from the optimality conditions that have been established for uncapacitated MDRPPs in Chapter 3.1.2. Since these conditions apply to the maximum number of times that edges are traversed in each *individual route* in an optimal solution, and they obviously apply to LARPs.

Therefore, optimality conditions O1, O2, O3 and O4 from Chapter 3.1.2 also apply for the MC-p-LARP, the MC-LARP, and the MM-p-LARP.

The optimality condition O8 that we introduce below is based on the number of facilities that can be opened on optimal solutions to the MC-p-LARP and the MC-LARP.

(O8) **MC-p-LARP and MC-LARP.** There exists an optimal solution in which every connected component of the graph induced by the edges that are used contains exactly one open facility.

Property O8 is obviously true for the MC-LARP. If some component of the graph induced by the edges used in an optimal solution contained more than one open facility, closing one of them would produce a solution with a better objective function value. In the case the MC-p-LARP a similar process will produce an alternative optimal solution.

When dealing with min-cost LARPs (with or without set-up costs), the fact that the number of operational depots is not known in advance allows us to prove that there exist optimal solutions in which no edge will traversed more than twice, provided that non-negative costs satisfy the triangle inequality, independently of whether or not the graph is complete. This is a very useful property, which makes a substantial difference with the case of MDRPPs stated on non-complete graphs, where edges can be traversed up to $2|D|$ times in optimal solutions, that we will exploit in some of the formulations that we propose for min-cost LARPs.

Proposition 5.1.3.

- *There exists an optimal MC-p-LARP solution in which no edge is traversed more than twice.*
- *There exists an optimal MC-LARP solution in which no edge is traversed more than twice.*

Proof. First we note that, since capacity constraints are not present, we can assume that only one route is carried out from each depot.

- Consider an optimal solution to a given MC-p-LARP in which an edge $e \in E$ is traversed by two routes T_1 and T_2, operating from two different open facilities, d_1, d_2. The solution obtained by merging T_1 and T_2 into a single route T, and arbitrarily closing one of the depots (for instance, d_2) is feasible for the MC-p-LARP, since the parity of the vertices does not change and the connectivity of the merged route with the remaining depot is guaranteed. Moreover, the merged solution is also optimal, since its routing cost has not changed. Edge e is traversed exactly twice in the merged route T, since otherwise two traversals of e could be removed, contradicting the optimality of the solution. This process can be repeated until all the routes traversing the same edge have been merged.

- For the MC-LARP we proceed as above, but now closing at each step the facility with the largest set-up cost. Moreover, the merged solution will have the same routing cost and smaller set-up costs.

5.2 Three-index variable formulations

We now present a linear integer formulation for the LARPs we have defined, which uses three-index variables, associated with the edges traversed in the routes of the open facilities. This three-index variable formulation can be adapted to all six LARPs defined in Chapter 5.1.

The formulation that we propose exploits the optimality conditions O2 and O3 to identify the set of edges E^y that can be traversed twice in an optimal solution. Recall that for the MC-p-LARP and the MC-LARP, E^y contains all the required edges plus the edges of T_C, whereas for the remaining models E^y contains all the required edges plus all edges connecting two distinct components.

Then, for each $e \in E$, let x_e^d be a binary variable indicating whether or not edge e is traversed by route from open facility d. For each $e \in E^y$, let y_e^d be a binary variable taking the value one if and only if edge e is traversed twice in the solution by route from facility d. For each $d \in D$, let z_d be a binary variable designating whether or not facility d is opened.

1. **MC-p-LARP**
   The MILP for the MC-p-LARP is as follows:

$$\text{minimize} \sum_{d \in D} \left(\sum_{e \in E} c_e x_e^d + \sum_{e \in E^y} y_e^d \right) \tag{5.1}$$

subject to

$$(x^d + y^d)(\delta(d)) \geq 2z_d \qquad d \in D \setminus V_R \tag{5.2}$$

$$(x^d + y^d)(\delta(S)) \geq 2x_e^d \qquad S \subseteq V \setminus \{d\},\ e \in E(S),\ d \in D \tag{5.3}$$

$$(x^d - y^d)(\delta(S) \setminus H) + y^d(H) \geq x^d(H) - |H| + 1 \qquad S \subset V,\ d \in D,\ H \subseteq \delta(S),\ |H| \text{ odd} \tag{5.4}$$

$$\sum_{d \in D} x_e^d = 1 \qquad e \in R \tag{5.5}$$

$$y_e^d \leq \sum_{d' \in D} x_e^{d'} \qquad e \in E^y,\ d \in D \qquad (5.6)$$

$$x_e^d \leq z_d \qquad e \in E,\ d \in D \qquad (5.7)$$

$$y_e^d \leq z_d \qquad e \in E^y,\ d \in D \qquad (5.8)$$

$$z(D) \leq p \qquad (5.9)$$

$$x_e^d \in \{0,1\} \qquad e \in E \qquad (5.10)$$

$$y_e^d \in \{0,1\} \qquad e \in E^y \qquad (5.11)$$

$$z_d \in \{0,1\} \qquad d \in D. \qquad (5.12)$$

Inequalities (5.2) ensure that if a potential location is opened, then there are at least two edges incident to it. Inequalities (5.3) are an adaptation of the well-known connectivity constraints, and ensure the connectivity of each route to its depot. This is guaranteed by imposing that if edge e is traversed by the route associated with facility $d \in D$, then the cutset of any vertex set containing the two end-nodes of e but not containing d must be crossed by at least two edges of that route. Inequalities (5.4) were presented and discussed in Chapter 3.3 for the MDRPPs and ensure the parity of every subset of vertices. Constraints (5.5) impose that all required edges be served by one route and (5.6) that no edge is traversed for the a second time unless it also has been traversed for a first time. By (5.7)–(5.8) no edge is traversed by the route of a facility that has not been opened. Inequality (5.9) means that at most p facilities are opened. The domains of the variables x, y and z are defined in constraints (5.10)–(5.12).

The above formulation contains $|E||D|$ x variables, $|E^y||D|$ y variables and $|D|$ z variables. There are $|D \setminus V_R|$ inequalities of type (5.2), $|R|$ inequalities (5.5), $|E^y||D|$ inequalities of type (5.6), $(|E| + |E^y|)\,|D|$ inequalities of types (5.7)–(5.8). The size of the families inequalities (5.3) and (5.4) is exponential in $|V|$.

2. **MC-LARP**

A formulation for the MC-LARP can be obtained from (5.2)–(5.12), by removing constraint (5.9), which limits the number of facilities to open, and adding the facilities set-up costs to the objective function, resulting in

$$\min \sum_{d \in D} f_d z_d + \sum_{d \in D} \sum_{e \in E} c_e x_e^d + \sum_{d \in D} \sum_{e \in E^y} y_e^d. \qquad (5.13)$$

3. **MM-p-LARP**

To formulate the minimization of the makespan is necessary to define a new variable w that represent the length of the longest route. Hence, the objective function becomes the minimization of w, subject to (5.2)–(5.12). Furthermore, a new family of constraints is needed, which relates the new variable w to the route lengths. These inequalities, also ensure that w represents the longest route:

$$w \geq \sum_{e \in E} c_e x_e^d + \sum_{e \in E^y} c_e y_e^d \quad v \in D. \tag{5.14}$$

4. **MC-p-LARP-UD**, **MC-LARP-UD** and **MM-p-LARP-UD**

   Dealing with the unit customer demands and the maximum number of customers to serve from each potential location b_d only requires adding to the corresponding uncapacitated formulation the following family of capacity constraints, one for each facility:

$$\sum_{e \in R} x_e^d \leq b_d z_d \quad v \in D. \tag{5.15}$$

5.2.1 Valid inequalities

We next introduce some families of valid inequalities that can be used to reinforce any of the formulations presented above.

- Since all vertices incident to required edges must be visited, for singletons $S = \{i\}$ with $i \in V_R$ the connectivity constraints (5.3) can be replaced with the tighter constraints

$$\sum_{d \in D} (x^d + y^d)(\delta(i)) \geq 2. \tag{5.16}$$

- The connectivity constraints (5.3) associated with components containing no potential facility can also be replaced with the tighter set of constraints. In particular for all $k \in K$ such that $V_k \cap D = \emptyset$, we have

$$\sum_{d \in D} (x^d + y^d)(\delta(V_k)) \geq 2. \tag{5.17}$$

- One can also impose logical conditions relating x variables associated with edges incident to potential facilities and z variables.

$$x_e^d + z_d + z_f \leq 2 \quad e = \{f, i\} \in E, d, f \in D, d \neq f. \tag{5.18}$$

  Constraints (5.18) mean that no edge incident with potential facility $f \in D$ can be traversed by the route of any other potential facility $d \in D$ if facility f is opened. When both end-nodes of edge e are potential facilities, i.e. $i = d \in D$, then inequality (5.18) can be reinforced to

$$\sum_{d' \in D} x_e^{d'} + z_d + z_f \leq 2 \quad e = \{f, d\} \in E, d, f \in D, \tag{5.19}$$

so that no edge connecting two potential facilities can be traversed by the route of any facility if both facilities are opened.

- In principle, only constraints (5.4) associated with singletons $S = \{v\}$ with $v \in V$, are needed to guarantee the parity of vertices in solutions. However, they are also valid for general vertex sets $S \subseteq V$. Imposing them for the general case leads to a formulation with a tighter LP relaxation. In fact, these inequalities can be further reinforced as we show below:

Proposition 5.2.1. *The inequality* (5.4) *associated with given* $d \in D$*,* $S \subset V$*,* $H \subseteq \delta(S)$*, with* $|H| \geq 3$ *odd, is dominated by the valid inequality*

$$(x^d - y^d)(\delta(S) \setminus H) + y^d(H) \geq x^d(H) - |H| + 2 - z_d. \tag{5.20}$$

Proof. Let $d \in D$, $S \subset V$, $H \subseteq \delta(S)$, with $|H| \geq 3$ and odd. To see that (5.20) is valid, recall that $z_d \in \{0, 1\}$ in any feasible solution. If $z_d = 0$, then $x^d_e = y^d_e = 0$, for all $e \in E$, so (5.20) reduces to $0 \geq -|H| + 2$, which holds by hypothesis. When $z_d = 1$, then (5.20) becomes (5.4). Indeed (5.20) are tighter than (5.4) since $2 - z_d \geq 1$. ■

Since the only inequalities (5.4) that are not dominated by the set (5.20) are those associated with odd edge sets $H \subset \delta(S)$ with $|H| < 3$, in the following we substitute the complete set of inequalities (5.4) by only its small family corresponding to singletons $S = \{v\}$ with $v \in V$, and subsets $H \subset \delta(S)$ consisting of just one edge, i.e. $|H| = 1$, plus the complete set of *reinforced parity constraints* (5.20).

5.2.2 Optimality condition for MC-p-LARP and MC-LARP

The optimality condition on the location variables O8 can be used to reinforce the three-index formulations for MC-p-LARP and MC-LARP. Modeling property O8 requires adding the following set of constraints:

$$z(V_k \cap D) \leq 1 \qquad k \in K \tag{5.21}$$

$$x^d_e = z_d \qquad e \in R_k,\ d \in D \cap V_k,\ k \in K \tag{5.22}$$

$$\sum_{d \in D \setminus V_k} x^d_e + z(V_k \cap D) = 1 \qquad k \in K \tag{5.23}$$

$$\sum_{d \in D \setminus V_k} x^d_e + z(V_k \cap D) \leq 1 \qquad e \in (E_k \setminus R_k) \cup \delta(V_k),\ k \in K \tag{5.24}$$

$$x^d_e \leq x^d_{e'} \qquad e \in \delta(V_k),\ e' \in R_k,\ k \in K,\ d \in D \setminus V_k. \tag{5.25}$$

By (5.21) at most one facility per component will be opened. Moreover, (5.22) ensure that if a facility is opened in a component, then all the required edges tin that component will be served from that facility. In its turn, (5.24) prevent any edge in the cut-set of a component where a facility is opened to be served from any facility located at any other component. The correct *propagation* of the route associated with an open facility is guaranteed by (5.25) together with the original set of constraints (5.7). In addition the following sets of inequalities can be used to reinforce the resulting formulation:

$$y_e^d \leq x_e^d \qquad e \in E^y, d \in D \tag{5.26}$$

$$\sum_{d \in D \setminus S} \left(x^d + y^d\right)(\delta(S)) \geq 2\,(1 - z(D \cap S)) \qquad S = \cup_{k \in K'} V_k, K' \subset K \tag{5.27}$$

$$z\,(V_k \cap V_{k'}) + \sum_{d \in (V_k \cap V_{k'})} x_e^d \leq 2 \qquad e \in \delta(V_k : V_{k'}), k, k' \in K, k \neq k'. \tag{5.28}$$

Inequalities (5.26) are a reinforcement of (5.6). They impose that if an edge is traversed twice, both traversals belong to the route of the same depot. The reinforced connectivity constraints (5.27) impose that if no open facility belongs to the group of components defining S, then the cutset of S must contain at least two edges of some route associated with a depot that does not belong to S. The set of inequalities (5.28) is generalization of (5.19).

5.2.3 Polyhedral analysis

In this section we study some properties of the polyhedron associated with the three-index formulation (5.1)–(5.12). In the following, the convex hull of vectors (x, y, z) with components in $[0, 1]$ that satisfy (5.2)–(5.9) is denoted by $P_{(MC-LARP)}$. In the proofs below we assume that there exists an edge connecting each pair of vertices. Similarly to Chapter 4.2.1, when such edges are *non-existing* in E, they represent, connecting given pairs of vertices, that only use *existing* edges of the set E. As in the previous chapter, using edges associated with such T-joins in the solutions that we will build, simplifies the presentation of the proofs, but has no effect on their. We also use $O \subseteq V$ to denote the set of R-odd vertices, and $O_k = O \cap V_k$, $k \in K$.

Proposition 5.2.2. $dim(P_{(MC-LARP)}) = |E||D| + |E^y||D| + |D| - |R|$ *if and only if every cut-edge set* $\delta(S) \subset V \setminus D$ *contains at least three edges, and every cut-edge set* $\delta(S)$ *such that* $S = \bigcup_{i \in K'} V_i \setminus D$, $\emptyset \neq K' \subset K$, *contains at least four edges.*

Proof. The condition is necessary. We follow the same idea as before for the MDRRP which, in turn, is based on Ghiani and Laporte (2000) for the RPP. To simplify the presentation, $e \in E$ and $e \in E^y$ are counted as two distinct edges.

- If there exists a cut-edge set with only one edge, then e should be a required edge and $\sum_{d \in D} x_e^d = 1$. Therefore, $P(MC-LARP) \subset \{x : \sum_{d \in D} x_e^d = 1\}$.

- 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 $P(MC-LARP) \subset \{x : \sum_{d \in D} x^d_{e^{(1)}} = 1$ and $\sum_{d \in D} x^d_{e^{(2)}} = 1\}$.
  - Otherwise, if $\delta(S)$ is R-even, $P(MC-LARP) \subset \{x : \sum_{d \in D} x^d_{e^{(1)}} = \sum_{d \in D} x^d_{e^{(2)}}\}$, and if $\delta(S)$ is R-odd, $P(MC-LARP) \subset \{x : \sum_{d \in D} x^d_{e^{(1)}} + \sum_{d \in D} x^d_{e^{(2)}} = 1\}$.
- Finally, there exists $S = \cup_{i \in K'} V_i \setminus D$, $\emptyset \neq K' \subset K$ with $\delta(S) = \{e^{(1)}, e^{(2)}, e^{(3)}\}$, then $P(MC-LARP) \subset \{x : \sum_{d \in D} x^d_{e^{(1)}} + \sum_{d \in D} x^d_{e^{(2)}} + \sum_{d \in D} x^d_{e^{(3)}} = 2\}$.

The condition is sufficient. Let us find $|E||D| + |E^y||D| + |D| - |R| + 1$ affinely independent solutions satisfying the connectivity, parity inequalities, associated with routes that start and terminate at the same open facility.

Consider a set of $|D|$ *reference solutions* $(\overline{x}(d), \overline{y}(d), \overline{z}(d))$, one associated with each potential facility $d \in D$. The reference solution associated with a given $d \in D$, consists of opening only facility $d \in D$, i.e. $\overline{z}(d)^d = 1$ and $\overline{z}(d)^{d'} = 0$ for all $d' \in D \setminus \{d\}$, together with a route carried out from d consisting of: (i) a traversal of all the required edges; (ii) one traversal of edge $\{v, d\}$ with $v \in O$ (this will be a second traversal for the required edges incident to d if both end-vertices are R-odd); and, (iii) two traversals of all the edges of T_C. By construction any reference solution is feasible.

A sufficiently large set of additional solutions, all of them affinely independent, can be obtained with slight modifications of the reference solutions. These *modified* solutions are *linked* both to the facilities of their corresponding reference solutions and to edges. We use the notation $(x(d,e), y(d,e), z(d,e))$, to denote the solution linked to the reference solution of facility $d \in D$ and edge $e \in E$. In particular, $x(d,e)^{d'}_{e'}$ denotes the component corresponding to the first traversal of edge $e' \in E$ in the route associated with facility d' in the solution linked to facility d and edge e. A similar notation will be used for the y and z components. When the reference solution and edge linked to a solution are clear from the context we will drop the parentheses and just write $x^{d'}_{e'}$. Let $d_0 \in D$ be an arbitrarily selected potential location. The set of affinely independent solutions linked with each potential facility $d \in D$ is defined below:

$a)$ For each non required edge $e = \{u, v\} \in E \setminus R$ we generate one or two solutions, depending on whether or not $e \in E^y$. In particular,

$a_1)$ if $e = \{u, v\} \notin E^y$, then we generate just one solution $(x(d,e), y(d,e), z(d,e))$ with $x^d_e = 1 - \overline{x}(d)^d_e$. Furthermore, to guarantee that the parity of u and v does not change we also set $x^d_{e_{\{u,r\}}} = 1 - \overline{x}(d)^d_{e_{\{u,r\}}}$, $x^d_{e_{\{v,r\}}} = 1 - \overline{x}(d)^d_{e_{\{v,r\}}}$. All other components remain as in the reference solution $(\overline{x}(d), \overline{y}(d), \overline{z}(d))$.

a_2) if $e = \{u, v\} \in E^y$, then e is one of the edges of T_C and $\overline{x}(d)_e^d = \overline{y}(d)_e^d = 1$. In this case we generate two new solutions $(x(d,e), y(d,e), z(d,e))$ and $(x'(d,e), y'(d,e), z'(d,e))$. For $(x(e,d), y(e,d), z(e,d))$, we keep $x_e^d = \overline{x}(d)_e^d = 1$ but set $y_e^d = 0$. To guarantee the parity of vertices u and v and the connectivity, the components corresponding to edges $e_u = \{d, u\}$ and $e_v = \{d, v\}$, take the value 1, i.e. $x_{e_u}^d = x_{e_v}^d = 1$. All other components remain as in the reference solution $(\overline{x}(d), \overline{y}(d), \overline{z}(d))$. For $(x'(d,e), y'(d,e), z'(d,e))$, we set $x_e'^d = y_e'^d = 0$, so the parity is not compromised. In contrast, the connectivity may be lost. To restore connectivity, it is enough to include the three edges connecting vertices u, v and the potential facility d via a triangle.

b) For each required edge $e = \{u, v\} \in R$ we generate one or two solutions, depending on whether or not $d = d_0$. In particular,

b_1) if $d = d_0$, then we generate just one solution $(x(d_0,e), y(d_0,e), z(d_0,e))$ with $x_e^{d_0} = \overline{x}(d_0)_e^{d_0} = 1$ and $y_e^{d_0} = 0$, for all $e = \{u, v\} \in R$. Furthermore, we set $x_{e_u}^{d_0} = 1 - \overline{x}(d_0)_{e_u}^{d_0}$ and $x_{e_v}^{d_0} = 1 - \overline{x}(d_0)_{e_v}^{d_0}$ where, as before, $e_u = (d_0, u)$ and $e_v = (d_0, v)$. This guarantees the parity of vertices u and v and the connectivity of $(x(d_0,e), y(d_0,e), z(d_0,e))$. All other components remain as in the reference solution $(\overline{x}(d_0), \overline{y}(d_0), \overline{z}(d_0))$.

b_2) if $d \neq d_0$, then we generate two new solutions: $(x(d,e), y(d,e), z(d,e))$ and $(x'(d,e), y'(d,e), z'(d,e))$, with one and two traversals of edge e, respectively. Solution $(x(d,e), y(d,e), z(d,e))$ is defined exactly as in item b_1). For $(x'(d,e), y'(d,e), z'(d,e))$ we open one additional potential facility $d' \neq d$, and define its associated route, taking into account that it is not possible to visit d' in the route from open facility d. For this d' is arbitrarily selected from $D \setminus \{d\}$ ensuring that is not an end-vertex of edge e. Then, we open both facilities d and d', i.e. $z'(d,e)^{d'} = z'(d,e)^d = 1$. Furthermore, associated with facility d, we set $x_e'^d = x_{e_u}'^d = x_{e_v}'^d = 1$, and all other x'^d and y'^d components at value zero. The first traversal of all other required edges is allocated to facility d'. That is, $x_{e'}'^{d'} = 1$ for all $e' \in R \setminus \{e, e_u, e_v\}$. To make consistent the route of facility d', we also allocate to d' one traversal of each edge connecting an R-odd vertex with facility d', plus two traversals of all the edges of T_C. All other components take the value 0.

The number of solutions defined in each of the items above is $|D|$ in the reference set, $(|E| - |E^y|)|D|$ in $a1$), $2(|E^y| - |R|)|D|$ in $a2$), $|R|$ in b_1), and $2|R|(|D| - 1)$ in b_2). Hence, we have found $|E||D| + |E^y||D| + |D| - |R|$ affinely independent solutions that satisfying the connectivity and parity inequalities. The remaining affinely independent solution consist of opening exactly p locations $d \in D$ and associate with each open location a consistent route, guaranteing that all required edges incident a potential location are allocated to that facility. Two traversals of the edges of

T_C can be arbitrarily allocated to the depots in order to guarantee the connectivity of the obtained solution. All the solutions considered are affinely independent, since each of the $|E||D| + |E^y||D| + |D| - |R|$ feasible solutions obtained in item a) and b) above contains at least one component with a different value from the values of that component in all other solutions. ■

Proposition 5.2.3. *The inequality $x_e^d \geq 0$, $e \in E$, $d \in D$, defines a facet of $P_{(MC-LARP)}$ if and only if every cut-set $\delta(S) \subset V \setminus D$ containing e contains at least four edges, every $\delta(S)$ such that $S = \bigcup_{i \in K'} V_i \setminus D$ ($\emptyset \neq K' \subset K$) contains at least five edges.*

Proof. The condition is necessary. The condition that every $\delta(S)$ such that $S = \bigcup_{i \in K'} V_i \setminus D$ ($\emptyset \neq K' \subset K$) has at least five edges is already necessary for the MDRPP when the set of available depots is fixed.

The condition is sufficient. If $e \in E \setminus R$, the face $\{x \in P(MC - LARP) : x_e^d = 0\}$ has the same dimension as the polytope associated with the MC-LARP defined on the graph obtained after removing edge e from G. Suppose now that $e \in R$. Observe that all the solutions obtained in the proof of Proposition 5.2.2 linked to depots $d' \in D$ different from d satisfy $x(d', e')_e^d = 0$, for all edges $e' \in E$. The number of such solutions depends on whether or not $d = d_0$. If $d \neq d_0$, this number is $(|D| - 1)(|E| + |E^y| + 1) - |R|$, whereas if $d = d_0$ this number will be $(|D| - 1)(|E| + |E^y| + 1)$. In order to generate additional solutions satisfying $x_e^d = 0$, affinely independent with the previous ones, let $d' \in D \setminus \{d\}$ be an arbitrarily selected potential location, and consider the solution $(\overline{x}, \overline{y}, \overline{z})$, where only facility d' is open, i.e. $\overline{z}_{d'} = 1$ and $\overline{z}_f = 0$ for all $f \neq d'$. The route of facility d' traverses all required ($\overline{x}_{e'}^{d'} = 1$, $e' \in R$), and contains one traversal of every edge $\{v, d'\}$ with $v \in O$, plus two traversals of all the edges of T_C. Then, for all $e' \in E$, we proceed as in the proof of Proposition 5.2.2 for generating solutions linked to facility d (all of them with $x(d, e')_e^d = 0$), using $(\overline{x}, \overline{y}, \overline{z})$ as reference solution. In this way, we will obtain $|E| + |E^y|$ solutions if $d \neq d_0$, or $|E| + |E^y| - |R|$ solutions when $d = d_0$. In both cases we have obtained $|D|(|E| + |E^y| + 1) - |R|$ affinely independent solutions that satisfy $x_e^d = 0$. ■

Proposition 5.2.4. *The inequality $x_e^d \leq 1$, $e \in E$, $d \in D$, induces a facet of $P_{(MC-LARP)}$ if and only if every cut-set $\delta(S)$ containing e contains 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 $\{x \in P_{(MC-LARP)} : x_e^d = 1\} \subset \{x \in P_{(MC-LARP)} : x_e^d = 1, x_f^d + x_g^d = 1\}$ if $\delta(S)$ is R-even, or $\{x \in P_{(MC-LARP)} : x_e^d = 1\} \subset \{x \in P_{(MC-LARP)} : x_e^d = 1, x_f^d - x_g^d = 0\}$ otherwise.

The condition is sufficient. Under the hypotheses, it is easy to show that there exist $|E||D|+|E^y||D|+|D|-|R|$ feasible and affinely independent solutions on the hyperplane $x_e^d = 1$. Let the first solution be solution $(\overline{x}, \overline{y}, \overline{z})$, where only facility d is open. Its associated route contains one traversal of all required edges $e \in R$, one traversal of each edge $\{v, d\}$ with $v \in O$, and two traversals of all the edges of T_C. In addition, if e does not belong to any of the previous sets of edges, then the route also traverses edge e, to guarantee that $\overline{x}_e^d = 1$, plus the two edges $e_u = \{d, u\}$ and $e_v = \{d, v\}$ to ensure parity. The remaining $|E||D| + |E^y||D| + |D| - |R| - 1$ solutions can be obtaining following a similar process to that applied in Proposition 5.2.2, where in each new solution one of the components is modified. ■

Proposition 5.2.5. *The connectivity inequality* (5.3) *associated with* $S = \bigcup_{i \in K'} V_i$ $(\emptyset \neq K' \subset K)$*,* $S \bigcap D = \emptyset$*,* $e \in E(S)$*, induces a facet of* $P_{(MC-LARP)}$ *if and only if the graphs induced by the connected components* $G(S)$ *and* $G(V \backslash S)$ *satisfy the following:* $i)$ $G(S)$ *is connected and each connected component of* $G(V \backslash S)$ *contains at least one open facility.* $ii)$ *For every subset of components in* $S' \subset S$ *(or* S' *in* $V \backslash S$*) with* $S' \bigcap D = \emptyset$*, the inequality* $|\delta(S') \backslash \delta(S)| \geq 2$*, holds.*

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 (5.3) associated with $G(S)$ is dominated by the connectivity inequality (5.3) corresponding to $G(S_1)$. A similar situation arises if some component of $G(V \backslash S)$ contains no open facility. 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 (5.3) associated with S' and $S \backslash S'$.

The condition is sufficient. It is easy to show that under the hypotheses, the set of $|E||D|+|E^y||D|+|D|-|R|$ affinely independent feasible solutions with $x_e^d = 1$ considered in the proof of Proposition 5.2.4 lie in the hyperplane $\sum_{e \in \delta(S)}(x_{e'}^d + y_{e'}^d) = 2x_e^d$. ■

Proposition 5.2.6. *The reinforced parity constraints* (5.4) *induce facets of* $P_{(MC-LARP)}$ *for* S *and* H *such that* $|\delta(S)| \geq |H| + 1$ *and* $H \cap \delta(D) = \emptyset$*.*

Proof. We first show that under the hypotheses, there exist $|E||D| + |E^y||D| + |D| - |R|$ affinely independent feasible solutions that satisfy the inequality as equality. For given sets S and H under the above conditions, let $d \in D$ be an arbitrarily selected potential facility and $v_k \in V_k$ an arbitrarily selected vertex in component $k \in K$. Let also $\widehat{e} \in \delta(S) \setminus H$ and $h_1 \in H$ be arbitrarily selected edges in their respective sets. Consider a feasible solution $(\overline{x}, \overline{y}, \overline{z})$ in which d is the only open facility and its associated route contains: $i)$ one traversal of each required edge $e \in R$; $ii)$ one traversal of each edge $\{v, v_k\}$ with $v \in O_k \setminus \{v_k\}$ (this will be a second traversal traversal for required edges with some R-odd end-vertex); iii) two traversals of all the edges of $T_C \setminus \delta(S)$ (edges with both end-vertices either in S or in $V \setminus S$); and iv) one traversal of edge $\widehat{e} \in \delta(S) \setminus H$ and of all $|H|$ edges of set H. By construction, $(\overline{x}, \overline{y}, \overline{z})$ is feasible and satisfies $(\overline{x}+\overline{y})(\delta(S)) = |H|+1$. The $|E||D|+|E^y||D|+|D|-|R|-1$ additional solutions are obtained from $(\overline{x}, \overline{y}, \overline{z})$, linked to the different edges $e \in E$ as explained next.

a) For all $e = \{u, v\} \notin H$, we proceed as in the proof of Proposition 5.2.2 and for each depot $d' \in D$, we obtain one or two points linked to edge e. The number of points that be obtain for each depot, depends on the case or subcase that applies to e depending on whether or not it belongs to R. In total we obtain D points if $e \in E \setminus R \setminus E^y$, $2|D|$ points if $e \in E^y$, and $2|D| - 1$ if $e \in R$.

b) For all $e = \{u, v\} \in H$ we define solutions $(x(d', e), y(d', e), z(d', e))$ linked to each $d' \in D$ and considered edge e, according to the following subcases:

- b_1) $e \in E \setminus E^y$ and $d' \neq d$. Then $\overline{x}_e^{d'} = \overline{y}_e^{d'} = 0$. We set $x_e^{d'} = 1$ and we use edges $e^u = \{d', u\}$ and $e^v = \{d', v\}$, so we set $x_{e^u}^{d'} = x_{e^v}^{d'} = 1$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$.
- b_2) $e \in E^y \setminus R$ and $d' = d$. Now $\overline{x}_e^d = 1$ and $\overline{y}_e^d = 0$. We set $y_e^d = 1$ and $x_{\widehat{e}}^d = 0$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$.
- b_3) $e \in E^y \setminus R$ and $d' \neq d$. We now generate two solutions: $(x(d', e), y(d', e), z(d', e))$ and $(x'(d', e), y'(d', e), z'(d', e))$. For the first solution, $(x(d', e), y(d', e), z(d', e))$, we set $x_e^{d'} = 1$ and traverse edges $e^u = \{d', u\}$ and $e^v = \{d', v\}$ in order to guarantee the parity. Hence, we set $x_{e^u}^{d'} = x_{e^v}^{d'} = 1$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$. For the second solution $(x'(d', e), y'(d', e), z'(d', e))$ we set $x_e'^{d'} = y_e'^{d'} = 1$. Now the parity is guaranteed but connectivity may be lost. To restore the connectivity, it is enough to include the three edges connecting vertices u, v and the potential facility d via a triangle.
- b_4) $e \in R$ and $d' \neq d$. Now $\overline{x}_e^d = 1$ and $\overline{y}_e^d = 0$. We generate two solutions: $(x(d', e), y(d', e), z(d', e))$ and $(x'(d', e), y'(d', e), z'(d', e))$. For the first solution, $(x(d', e), y(d', e), z(d', e))$, we set $x_e^{d'} = 1$ and use edges $e^u = \{d', u\}$ and $e^v = \{d', v\}$. Thus, we set $x_{e^u}^{d'} = x_{e^v}^{d'} = 1$. We also set $x_e^d = x_{\widehat{e}}^d = 0$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$. For $(x'(d', e), y'(d', e), z'(d', e))$ we set $x_e'^{d'} = y_e'^{d'} = 1$. We also set $x_e'^d = x_{\widehat{e}}'^d = 0$. Parity is guaranteed although connectivity may be lost. To restore it, it is enough to include the three edges connecting vertices u, v and the potential facility d via a triangle.

Furthermore, when $e \neq h_1$ we generate the following additional points linked to depot d and edge e, $(x(d, e), y(d, e), z(d, e))$, according to the following subcases:

b'_1) $e \in E \setminus E^y$. We set $x_e^d = x_{\hat{e}}^d = 0$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$.

b'_2) $e \in E^y \setminus R$. We set $x_e^d = x_{\hat{e}}^d = 0$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$.

b'_3) $e \in R$. We set $x_e^d = y_e^d = 1$ and $x_{\hat{e}}^d = 0$. All other components remain as in $(\overline{x}, \overline{y}, \overline{z})$.

In total we have generated $|E||D| + |E^y||D| + |D| - |R|$ feasible solutions, all of which satisfy the inequality (5.4) associated with S and H as equality. The result follows, since all points are affinely independent. ■

5.3 Two-index variable formulations

Similar to MDRPPs alternative modeling option to the one presented in the previous section is to work with two-index variable formulations that aggregate the information of all the routes. As before, such models are only valid for problems in which the objective is an aggregate measure of all routes (MC-p-LARP and MC-LARP), and the feasibility of the solutions can be derived from the aggregated information. Therefore they are not valid if the objective is to minimize the makespan, which reflects the cost of one specific route, or for problems with capacity constraints, where the arcs traversed by each of the routes need to be known.

Next we propose two-index variable formulation valid for MC-p-LARP and MC-LARP. The formulations exploits Proposition 5.1.3: regardless of whether or not G is a complete graph, there exists an optimal solution to both MC-p-LARP and MC-LARP in which no edge is traversed more than twice. Therefore, in both cases the total number of traversals of each edge can be represented by means of only two binary variables, one for the first one and one for the second one. Since connectivity and parity conditions are not sufficient to guarantee that the routes start and end at the same facility. We introduce an extension of the set of constraints (4.5) proposed for the MDRPP, which now integrate locational decision variables as well.

We use the same location variables as above so the binary variable z_d, $d \in D$, indicates whether or not a facility is established at d. As for the routing variables, let x_e denote the binary variable for the first traversal of edge $e \in E$, and y_e the binary variable indicating whether or not edge $e \in E^y$ is traversed a second time.

5.3.1 Return-to-facility constraints

Before presenting the formulation we discuss the Rt-FCs which guarantee that all routes start and end at the same facility. These are an extension of the Rt-

FCs for the MDRPPs, which highlight that the set of open facilities involved in the edges of the cut-sets is relevant to guarantee consistent routes in LARPs. It is important to note that the Rt-FCs introduced for MDRPPs are no longer valid for LARPs since they assume that the set of depots from which routes originate is known. However, since the set of potential locations that will actually become depots for the routes is not known in advance for LARPs, location variables are required in the proposed inequalities. As we will see, the resulting inequalities are quite involved.

In Figure 5.2 the gray squares represent potential facilities and the solid lines correspond to required edges. This figure illustrates not only that connectivity and parity constraints are not sufficient to guarantee well-defined routes, but also that the conditions needed to guarantee consistent routes in LARPs necessarily depend on the set of open facilities. Observe that if only one or two of the three potential locations opened, the displayed solution would be feasible and, depending on the case, it would consist of one or two well-defined routes. Instead, if all three potential facilities opened, the displayed solution would be infeasible since it is not possible to decompose it into three routes, each starting and ending at the same facility. Moreover, if all three potential facilities opened, any feasible solution should have at least three more edges (or additional traversals of the existing edges) in the cut-set of $S = \{1, 2\}$. This idea is formalized below.

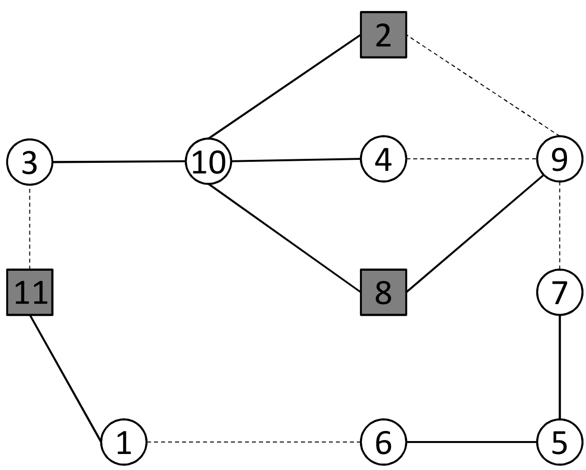


Figure 5.2: Infeasible solution satisfying connectivity and parity constraints

Consider a vertex set $S \subset V \setminus D$ and a subset of potential facilities $D' = \{d_1, \ldots, d_r\} \subset D$. Consider also a subset of edges $H \subset \delta(S) \cap \delta(D')$. Denote by $H_i \neq \emptyset$ the set of edges of H incident with facility $i \in D'$ and assume that each H_i contains an odd number of edges. Finally, partition $\delta(S) \setminus H$ in the following to sets: $F^{D'}_{S,H} = (\delta(S) \setminus H) \setminus \delta(D \setminus D')$, the set of edges of $\delta(S) \setminus H$

that are not incident to any potential facility different from those of D', and $Q^{D'}_{S,H} = (\delta(S) \setminus H) \cap \delta(D \setminus D')$, the set of edges of $\delta(S) \setminus H$ incident to some potential facility not in D'. The inequality that we propose contains bilinear terms that will be discussed and linearized later on.

Proposition 5.3.1. *The RtFC*

$$(x-y)\left(F^{D'}_{S,H}\right) + \sum_{d \in D \setminus D'} (1-z_d)(x-y)\left(Q^{D'}_{S,H} \cap \delta(d)\right) + y(H) \geq x(H) - |H| + z(D') \tag{5.29}$$

associated with S, D', and H as defined above is valid for MC-p-LARP and MC-LARP.

Proof. Let (z, x, y) be a feasible LARP solution and note that the RtFC (5.29) is only active if $x(H) - |H| + z(D') > 0$. Since $x(H) - |H| \leq 0$, a necessary condition is that $z(D') \geq 1$. Consider the following cases:

a) $x(H) = |H|$ and $z(D') \geq 1$. The right-hand side of the RtFC reduces to $z(D') \geq 1$. Since $x(H) = |H|$, then $x_e = 1$, for all $e \in H_i$, $i \in \{1, \ldots, r\}$. Given that all the edges in each H_i are incident with the same potential facility and $|H_i|$ is odd, there must be at least one additional traversal of some edge in the cut-set associated with each open facility of the set D'. That is, in total $z(D')$ additional traversals are needed, which must correspond either to second traversals of edges in H (term $y(H)$), or to first traversals of edges in $\delta(S) \setminus H$. In the latter case, the first traversal may correspond to edges not incident with potential locations of $D \setminus D'$, represented by the first term of the left-hand side $(x-y)\left(F^{D'}_{S,H}\right)$, or to potential locations of $D \setminus D'$, provided that the involved potential locations are not open, represented by the second term of the left-hand side $\sum_{d \in D \setminus D'} (1-z_d)(x-y)\left(Q^{D'}_{S,H} \cap \delta(d)\right)$. The bilinear terms are necessary since the edges incident with potential locations in $D \setminus D'$ may contribute to the overall count only when the potential facility involved remains closed.

b) $x(H) = |H| - 1$ and $z(D') \geq 2$. The right-hand side of the RtFC reduces to $z(D') - 1 \geq 1$. In this case exactly one of the edges of H is not traversed in solution (z, x, y). In full, let us assume that $x(H_1) = |H_1| - 1$ (which is even), and $x(H_i) = |H_i|$, $i \in \{2, \ldots, r\}$. Consider now $\overline{D'} = D' \setminus \{d_1\}$, and $\overline{H} = H \setminus H_1$.

b_1) The RtFC associated with S, $\overline{D'}$, and $\overline{H}$ corresponds to case a), since $x(H \setminus H_1) = (|H \setminus H_1|)$ and $z(\overline{D'}) = z(D' \setminus \{d_1\}) \geq 1$. Therefore it is valid.

b_2) The RtFC associated with S, D', and H is dominated by the RtFC associated with S, $\overline{D'}$, and $\overline{H}$. Both inequalities have the same right-hand side, and the left-hand side of the former is weaker than the right-hand side of the later since $y(H) \geq y(\overline{H})$ and $(x-y)\left(F_{S,H}^{D'}\right) \geq (x-y)\left(F_{S,\overline{H}}^{\overline{D'}}\right) + (1-z_{d_1})(x-y)\left(((\delta(S) \setminus H) \cap \delta(d_1)\right)$.

Hence, the RtFC associated with S, D', and H is valid.

c) $x(H) = |H| - 2$ and $z(D') \geq 3$. The right-hand side of the RtFC is $z(D') - 2 \geq 1$. There are exactly two edges of H, say e_1, e_2 that are not traversed in the solution (z, x, y). Consider the two possible subcases:

c_1) $e_1, e_2 \in H_1$. Then, quite similarly to case b, the RtFC associated with S, D', and H is dominated by that associated with S, D', $\overline{H} = \{\overline{H_1}, H_2, \ldots, H_r\}$, with $\overline{H_1} = H_1 \setminus \{e_1, e_2\}$, which corresponds to case a).

c_2) e_1 and e_2 are incident with two different depots, i.e. $e_1 \in H_1$, $e_2 \in H_2$. Then, the RtFC associated with S, D', and H is dominated by that associated with S, $\overline{H} = H \setminus \{H_1, H_2\}$ and $\overline{D}' = D' \setminus \{d_1, d_2\}$ which also corresponds to case a).

Hence, the RtFC associated with S, D', and H is valid.

d) All other cases can be handled similarly. ■

For illustrative purposes, consider again the solution depicted in Figure 5.2 with two alternative values for the location variables: one where all three potential facilities are open, i.e. $z_{L1}^1 = z_{L2}^1 = z_{L3}^1 = 1$ which, as explained, is infeasible, and another one where only $L1$ and $L2$ are open, i.e. $z_{L1}^2 = z_{L2}^2 = 1, z_{L3}^2 = 0$, which is feasible. Consider the vertex set $S = \{1, 2\}$, $H_1 = \{(1, L1)\}$ and $H_2 = \{(2, L2)\}$. In both cases let $D' = \{L1, L2\}$, so $F_{S,H}^{D'} = \{(2, 3)\}$ and $Q_{S,H}^{D'} = \{(2, L3)\}$.

For the infeasible solution z^1 we have $z^1(D') = z_{L1}^1 + z_{L2}^1 = 2$. Since $z_{L3}^1 = 1$ we also have $h_{(2,L3)}^{d_3} = 0$, so $\sum_{d \in D \setminus D'} \overline{h}^d \left(Q_{S,H}^{D'} \cap \delta(d)\right) = 0$. Therefore, the associated RtFC (**??**) is violated since $x(H) - |H| + z(D') = 2$, but $(x-y)\left(F_{S,H}^{D'}\right) + \sum_{d \in D \setminus D'} h^d \left(Q_{S,H}^{D'} \cap \delta(d)\right) + y(H) = 1$.

If we instead consider the feasible solution z^2, we also have $z^2(D') = z_{L1}^2 + z_{L2}^2 = 2$, but now $h_{(2,L3)}^{d_3} = 1$, since $z_{L3}^2 = 0$. Hence, $\sum_{d \in D \setminus D'} \overline{h}^d \left(Q_{S,H}^{D'} \cap \delta(d)\right) = 1$ and the left-hand side of the RtFC is becomes 1 + 1, which coincides with the value of the right-hand side that does not change. Hence, as expected, the

RtFC is not violated for this feasible solution.

In order to integrate the set of inequalities (5.29) within a MILP formulation it is necessary to linearize the bilinear terms that they include. For this we define additional decision variables representing the products $h_{ed} = (1 - z_d)(x_e - y_e)$ for the edges $e \in \delta(d)$, with $d \in D$. These variables will take the value 1 if and only if edge e, which is incident with potential facility d, is traversed exactly once and the facility located at d is not open. Observe that the number $\sum_{d \in D} |\delta(d)|$ of new variables is very moderate since we are assuming that $|V_k \cap D| \leq 1$, for all $k \in K$. This number is clearly smaller than the number of two-index variables. The new set of variables h and variables x, y and z can be related with the usual *linearizing* constraints:

$$h_{ed} \leq (1 - z_d) \qquad d \in D, e \in \delta(d) \tag{5.30}$$

$$h_{ed} \leq (x_e - y_e) \qquad d \in D, e \in \delta(d) \tag{5.31}$$

$$(1 - z_d) + (x_e - y_e) \leq 1 + h_{ed} \qquad d \in D, e \in \delta(d). \tag{5.32}$$

5.3.2 MILP formulation for MC-p-LARP and MC-LARP

The MILP for the MC-p-LARP is presented below:

$$\text{minimize} \sum_{e \in E} c_e x_e + \sum_{e \in E^y} y_e \tag{5.33}$$

subject to

$$(x + y)(\delta(d)) \geq 2z_d \qquad d \in D \setminus V_R \tag{5.34}$$

$$(x + y)(\delta(S)) \geq 2(1 - z(S)) \qquad S \subseteq V, S \cap V_R \neq \emptyset \tag{5.35}$$

$$(x - y)(\delta(S) \setminus H) + y(H) \geq x(H) - |H| + 1 \qquad S \subset V,\ H \subseteq \delta(S),\ |H| \text{ odd} \tag{5.36}$$

$$(x - y)\left(F^{D'}_{S,H}\right) + \sum_{d \in D \setminus D'} h^d \left(Q^{D'}_{S,H} \cap \delta(d)\right) + y(H) \geq x(H) - |H| + z(D') \tag{5.37}$$

$$D' = \{d_1, \dots, d_r\} \subset D,\ S \subset V \setminus D,\ H = H_1 \cup \dots H_r,\ H_i \subseteq \delta(S) \cap \delta(d_i),\ |H_i| \text{ odd},\ i = 1, \dots, r,\ r > 1$$

$$x_e = 1 \qquad e \in R \tag{5.38}$$

$$y_e \leq x_e \qquad e \in E^y \tag{5.39}$$

$$z(D) \leq p \tag{5.40}$$

$$h_{ed} + z_d \leq 1 \qquad d \in D, e \in \delta(d) \tag{5.41}$$

$$h_{ed} + y_e \leq x_e \qquad d \in D, e \in \delta(d) \tag{5.42}$$

$$x_e \leq z_d + y_e + h_{ed} \qquad d \in D, e \in \delta(d) \tag{5.43}$$

$$x_e \in \{0, 1\} \qquad e \in E \tag{5.44}$$

$$y_e \in \{0, 1\} \qquad e \in E^y \tag{5.45}$$

$$z_d \in \{0, 1\} \qquad d \in D \tag{5.46}$$

$$h_{ed} \in \{0,1\} \qquad d \in D, e \in \delta(d). \tag{5.47}$$

Inequalities (5.34) ensure that open facilities are used and the family (5.35) is an adaptation of the well-known connectivity inequalities: there must be at least two edge traversals in the cut-set of a given set of vertices S containing no open facility whenever S contains some vertex that must be visited. Inequalities (5.36) have a similar explanation to that of (5.4) and ensure the parity (even degree) of every subset of vertices. They have been used in the two-index formulation for the MDRPP proposed in Chapter 4 (observe that they do not involve any location variable). The RtFCs (5.37) have been discussed above. Equalities (5.38) ensure that all required edges are served whereas constraints (5.39) mean that an edge cannot be traversed for a second time unless it also has been traversed for the first time. The limit on the maximum number of facilities that can be opened is imposed by (5.40). The linearization of the set of new variables h and its relation to the other decision variables is given in (5.41)–(5.43). Finally, the domains of the different sets of decision variables are stated in (5.44)–(5.47).

The above formulation contains $|E|$ x, $|E^y|$ y, and $|D|$ z variables. As mentioned, the number of h variables is $\sum_{d\in D}|\delta(d)|$. There are $|D \setminus V_R|$ inequalities of type (5.34), $|R|$ inequalities (5.38), $|E^y|$ inequalities of type (5.39). The number of constraints in each family (5.41)–(5.43) is $\sum_{d\in D}|\delta(d)|$. The number of inequalities (5.35), (5.36), and (5.37) is exponential in $|V|$.

- **MC-LARP**
  Since the domains of MC-p-LARP and MC-LARP are the same, except for constraint (5.40) on the maximum number of open facilities, in order to adapt the above formulation to the MC-LARP, we only need to discard this constraint and to update the objective function to

$$\min \sum_{d\in D} f_d z_d + \sum_{e\in E} c_e x_e + \sum_{e\in E^y} y_e. \tag{5.48}$$

Proposition 5.3.2. *Formulation* (5.34)–(5.47) *is valid for the MC-p-LARP and for the MC-LARP.*

Proof. By Proposition 5.3.1 inequalities (5.37) are valid. Therefore, if a solution $(\overline{x},\overline{y},\overline{z})$ is feasible for the MC-p-LARP or the MC-LARP no violated inequality of this family exists. We now show that if a solution $(\overline{x},\overline{y},\overline{z})$ satisfying (5.34)–(5.36), (5.38)–(5.40), and (5.44)–(5.45) is not feasible for the MC-p-LARP or the MC-LARP, then there exists a constraint (5.37) violated by the solution. Because of the connectivity and parity constraints (5.35)–(5.36), if $(\overline{x},\overline{y},\overline{z})$ is not feasible then in any decomposition of the solution in edge-disjoint simple tours, there is one simple tour T traversing at least two open facilities. Let $d_1, d_2 \in D$ be two open facilities that are consecutive in the tour T, and let $P_{d_1 d_2}$ the subpath of T connecting d_1, and d_2 and $S^T = V(P_{d_1 d_2}) \setminus D$.

- If the decomposition contains no simple tour T' incident with some vertex of S^T, i.e., $S^T \cap V(T') \neq \emptyset$, then the RtFC (5.37) associated with $S = S^T$, $D' = \{d_1, d_2\}$, $H_1 = S \cap \delta(d_1)$, $H_2 = S \cap \delta(d_2)$, $F_{S,H}^{D'} = (\delta(S) \setminus H) \setminus \delta(D \setminus D')$ and $Q_{S,H}^{D'} = (\delta(S) \setminus H) \cap \delta(D \setminus D')$ is violated by $(\overline{x}, \overline{y}, \overline{z})$, since all the terms in the left-hand side of (5.37) take the value zero, but the right-hand side takes the value two, since $\overline{z}_{d_1} = \overline{z}_{d_2} = 1$.

- Suppose now that the decomposition contains a simple tour T' incident with some vertex of S. Let $\{v\} \in S \cap V(T')$ (arbitrarily selected, if there is more than one such vertex). Consider the following subcases:

  - T' does not intersect with $V(T) \setminus P_{d_1 d_2}$. Consider $S^{T'}$ consisting of all vertices of $V(T')$ which are not open facilities in $\overline{z}$ (possibly all $V(T')$). Then the RtFC (5.37) associated with $S = S^T \cup S^{T'}$, $D' = \{d_1, d_2\}$, $H_1 = S \cap \delta(d_1)$, $H_2 = S \cap \delta(d_2)$, $F_{S,H}^{D'} = (\delta(S) \setminus H) \setminus \delta(D \setminus D')$ and $Q_{S,H}^{D'} = (\delta(S) \setminus H) \cap \delta(D \setminus D')$ is violated by $(\overline{x}, \overline{y}, \overline{z})$. Again all the terms in the left-hand side of (5.37) take the value zero,but the right-hand side takes the value two.
  - T' intersects with $V(T) \setminus P_{d_1 d_2}$. Let $\{v'\} \in V(T') \cap (V(T) \setminus P_{d_1 d_2})$. If several such vertices exist v' the *first* vertex *after* d_2 following the same orientation as that of $P_{d_1 d_2}$. Observe that now T' must traverse some open facility, say $d' \in D \cup V(T')$, different from those of $\{d_1, d_2\}$. Otherwise a different decomposition of the solution of simple tours would exist, where d_1 and d_2 are no longer consecutive open facilities in the same simple tour. Consider now the subpaths of T', $P_{v,v'}$ and $P_{v,d'}$, and define $S^{T'} = V(P_{v,v'}) \cup \big(V(P_{v,d'}) \setminus D\big)$. Then, the RtFC (5.37) associated with $S = S^T \cup S^{T'}$, $D' = \{d_1, d_2\}$, $H_1 = S \cap \delta(d_1)$, $H_2 = S \cap \delta(d_2)$, $F_{S,H}^{D'} = (\delta(S) \setminus H) \setminus \delta(D \setminus D')$ and $Q_{S,H}^{D'} = (\delta(S) \setminus H) \cap \delta(D \setminus D')$ is violated by $(\overline{x}, \overline{y}, \overline{z})$. Now the left-hand side of (5.37) takes the value one (corresponding to the last edge of the path $P_{v,v'}$, but the left-hand side is two. ■

Remark 5.3.1. An additional consequence of the above proof is that RtFC inequalities (5.37) associated with subsets D' with two depots suffice to guarantee that the proposed formulation is valid.

Modeling optimality condition O8 for the two-index formulations is not easy. In fact, we do not know how to impose this condition without incorporating additional decision variables, and preliminary experiments clearly indicate that such an alternative would not be competitive with the original formulations.

5.3.3 Valid inequalities

Some of the valid inequalities presented in Chapter 5.2.1 can be adapted to reinforce the formulation above. In particular, the reinforced connectivity inequalities (5.16) associated with singletons that must be visited $S = \{i\}$ with $i \in V_R$ can be expressed in terms of the aggregated x and y variables as

$$(x + y)(\delta(i)) \geq 2. \tag{5.49}$$

Analogously, (5.17) can be expressed in terms of the aggregated x and y variables to reinforce constraints (5.35) associated with components containing no potential facility as

$$(x + y)(\delta(V_k)) \geq 2. \tag{5.50}$$

Finally, the logical relation between the z and x variables associated with edges connecting two facilities (5.18) can be rewritten as

$$x_e + z_d + z_f \leq 2 \quad e = \{d, f\} \in \gamma(D). \tag{5.51}$$

5.4 Branch-and-cut algorithm

We have developed an exact branch-and-cut algorithm to solve each of the models presented in this chapter, based on the formulations proposed above. The overall solution algorithm is similar for three- and two-index formulations. As usual, we initially relax the families of constraints of exponential size. After each LP iteration these are then separated to detect whether or not there are constraints of any of these families violated by the current LP solution. If so, the detected violated constraints are incorporated in the current formulation, and the reinforced formulation is solved.

5.4.1 Initial relaxation

The algorithm starts with all integrality conditions relaxed and only a subset of constraints. In the initial formulations we include all non-exponential sets of constraints, plus a small subset of connectivity and parity inequalities. More precisely, the initial connectivity constraints considered are associated with the singletons that must be visited, i.e. $S = \{i\}$, $i \in V_R$, and with the components that contain no potential facility, i.e. $S = V_k$, $k \in K$, with $V_k \cap D = \emptyset$. The initial set of parity constraints is restricted to those associated with R-odd singletons. That is, for the three-index formulations, constraints (5.3) are initially replaced with (5.16)–(5.17) and the only parity constraints initially included are the inequalities (5.4) associated with R-odd singletons $S = \{v\}$, with $|\delta_R(v)|$ odd.

Furthermore, all logical inequalties (5.18) and (5.19) are added. For the two-index formulations, constraints (5.35)–(5.37) are initially replaced with (5.49)–(5.50), the only parity constraints (5.36) initially included are those associated with R-odd singletons $S = \{v\}$ with $|\delta_R(v)|$ odd, and all logical inequalities (5.51) are added.

RtFCs (5.37) are handled as lazy constraints, so they are only separated at the nodes with an integer LP solution. In contrast, all other families of relaxed inequalities are separated whenever the current LP solution is fractional. We then first apply a heuristic separation and only resort to the exact separation when the heuristic fails in finding any violated cut. Below we detail the separation procedures that are applied in each case.

5.4.2 Separation of inequalities for the three-index formulations

Let $(\overline{x}, \overline{y}, \overline{z})$ denote the current LP solution and let $G(\overline{x}, \overline{y})$ be the support graph associated with $(\overline{x}, \overline{y})$ at any iteration of the algorithm. For each facility $d \in D$, we denote by $(\overline{x}^d, \overline{y}^d)$ the partial LP solution associated with the potential facility d and by $G^d_{\overline{x},\overline{y}} = (V^d, E_{\overline{x}^d,\overline{y}^d})$ its corresponding support graph, which can be obtained from G by eliminating all edges in E with $\overline{x}^d_e = 0$ and all vertices that are not incident with any edge of $E_{\overline{x}^d,\overline{y}^d}$.

Separation of the connectivity constraints (5.3)
For each potential facility $d \in D$, we check whether $G^d_{\overline{x},\overline{y}}$ is connected. If not, each connected component C with vertex set $V(C) \subseteq V^d \setminus \{d\}$ defines a violated connectivity constraint (5.3). When the current LP solution is integer, i.e. $\overline{z}_d = 1$, the above separation procedure is exact. However, when the current LP solution is fractional, it may fail to find a violated constraint (5.3) even if one exists. Therefore, when $G^d_{\overline{x},\overline{y}}$ contains one single connected component we search for connected components in the subgraph of $G^d_{\overline{x},\overline{y}}$ that contains only those edges with values $\overline{x}^d_e + \overline{y}^d_e \geq \varepsilon$, where ε is a given parameter. We then compute the current value of $(\overline{x}^d_e + \overline{y}^d_e)(V(C))$ for each connected component C with vertex set $V(C) \subseteq V^d \setminus \{d\}$. If for some edge $e \in \gamma(V(C))$ the inequality $(\overline{x}^d + \overline{y}^d)\,(\delta(V(C))) < 2x^d_e$ is satisfied, then the connectivity inequality (5.3) associated with $V(C)$ is violated by $(\overline{x}^d, \overline{y}^d)$. Finally, if no violated constraint has been found with the above heuristic, we build the tree of min-cuts T^d of $G^d_{\overline{x},\overline{y}}$ with capacities given by $\overline{x}^d_e + \overline{y}^d_e$. For each edge $e = \{u, v\}$ in $E_{\overline{x}^d,\overline{y}^d}$ with $u, v \neq d$, the minimum cut $\delta(S)$ such that $e \in \gamma(S)$ is easily obtained from the min-cut tree T^d. If the value of the min-cut is smaller than $2\overline{x}^d_e$, then the inequality (5.3) associated with S and d is violated by $(\overline{x}^d, \overline{y}^d)$. The above separation procedure is exact and similar to that applied in Chapter 3.4.2 to the connectivity constraints of the three-index formulation for the MDRPPs.

Separation of the parity inequalities (5.20)
Since the initial formulation includes all parity constraints (5.4) associated with singletons, for integer solutions $(\overline{x}, \overline{y}, \overline{z})$ the reinforced parity inequalities (5.20)

are always satisfied. When $(\overline{x}, \overline{y}, \overline{z})$ is not integer, we first apply a heuristic and we only resort to the exact separation if the heuristic fails. The heuristic and exact method for inequalities (5.20) are adaptations of those applied in Chapter 3.4.2 to the simple parity constraints (5.4) of the three-index formulation for the MDRPPs, where now the right-hand side of the inequality is $2 - z_d$, instead of 1.

Concerning the heuristic for each potential facility $d \in D$, we find the connected components of the subgraph $G^d(\overline{x}, \overline{y})$ induced by edges with values $b_e^d = \min\{(\overline{x}_e^d - \overline{y}_e^d), 1 - (\overline{x}_e^d - \overline{y}_e^d)\} > \varepsilon$, where ε is a given parameter. Then, if $S \subset V$ is the vertex set of one of the components, its associated edge set is $H = \{e \in \delta(S) \mid 1 - (\overline{x}_e^d - \overline{y}_e^d) < \overline{x}_e^d - \overline{y}_e^d\}$. If $b^d(\delta(S)) < 2 - \overline{z}_d$ and $|H|$ is odd, then the parity constraint (5.20) associated with S and H is violated by $(\overline{x}^d, \overline{y}^d, \overline{z}^d)$. If $|H|$ is even, we obtain an odd set $|H|$ by either removing one edge from $|H|$ (and transferring it to $\delta(S) \setminus H$) or by adding to H one edge currently in $\delta(S) \setminus H$. Again, the smallest increment is obtained with

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

Then, if $b^d(\delta(S)) + \Delta < 2 - \overline{z}_d$, the parity constraint (5.20) associated with S and the updated set H is violated by $(\overline{x}^d, \overline{y}^d)$. Otherwise, the heuristic fails to find a constraint violation.

The exact method constructs, for each $d \in D$, the tree of min-cuts T^d of the support graph G^d with capacities b^d. When T^d has a cut $\delta(S)$ of capacity smaller than $2 - \overline{z}_d$, i.e. $b(\delta(S)) < 2 - \overline{z}_d$, we consider its vertex set S, and the set of edges $H = \{e \in \delta(S) \mid (\overline{x}_e^d - \overline{y}_e^d) \geq 0.5\}$. If $|H|$ is odd, then H defines, together with S, a violated inequality of type (5.20). Otherwise, if $|H|$ is even, we update the set H to an odd set by moving an edge as mentioned above. When $b^d(\delta(S)) + \Delta < 2 - \overline{z}_d$, the updated set H defines a violated inequality (5.20) for d and S for the current solution $(\overline{x}^d, \overline{y}^d)$.

5.4.3 Separation of inequalities for the two-index formulations

Let $G(\overline{x}, \overline{y})$ denote the support graph associated with the LP solution $(\overline{x}, \overline{y}, \overline{z})$ at any iteration of the algorithm.

Separation of the connectivity inequalities (5.35)
The separation of constraints (5.35) is an adaptation of the procedure presented in Chapter 4.3.2 for the connectivity constraints of the two-index formulation for the MC-MDRPP. Now we need to take into account that the right-hand side is $2\,(1 - z(S))$ instead of 2. We first check whether $G(\overline{x}, \overline{y})$ is connected. If not, the vertex set of any component containing no depot defines a violated cut. As before, when $(\overline{x}, \overline{y}, \overline{z})$ is integer the above separation is exact, but it may fail for fractional solutions. In such a case, the connected components are identified in the subgraph of $G(\overline{x}, \overline{y})$ with only those edges with $\overline{x}_e + \overline{y}_e \geq \varepsilon$, where ε is a

given parameter. Then, the value $(\overline{x}+\overline{y})\,(\delta(V(C)))$ is computed for each component $V(C)$ and compared to $2(1-\overline{z}(S))$. If $(\overline{x}+\overline{y})\,(\delta(V(C))) < 2(1-z(S))$, the constraint (5.35) associated with $V(C)$ is violated by $(\overline{x},\overline{y},\overline{z})$.

For the exact separation we build the tree of min-cuts of $G(\overline{x},\overline{y})$ with capacities given by $\overline{x}_e+\overline{y}_e$, and look for min-cuts $\delta(S)$ of value $(\overline{x},\overline{y})\,(\delta(S)) < 2$. When $(\overline{x},\overline{y})\,(\delta(S)) < 2(1-z(S))$, then the inequality (5.35) associated with S is violated by $(\overline{x},\overline{y},\overline{z})$.

Separation of the parity inequalities (5.36)
We use the separation of constraints (5.36) presented in Chapter 4.3.2 for the two-index formulation for the MC-MDRPP. Since the initial formulation includes the inequalities associated with singletons, we only separate them at fractional solutions. We first apply the heuristic and only resort to the exact separation if it fails.

Separation of the return-to-facility inequalities (5.37) RtFCs (5.37) are handled as lazy constraints, so they are only separated when the LP solution $(\overline{x},\overline{y},\overline{z})$ is integer. In such a case violated inequalities can be easily identified by first finding a tour decomposition of the current solution and then checking whether any of the tours contains a path $P_{d_1d_2}$ connecting two (consecutive) open facilities. If so, $D' = \{d_1, d_2\}$ and $S = V(P_{d_1d_2}) \setminus D'$ defines a violated cut.

5.5 Computational Experience

In this section we present the results of the computational experience we have conducted to assess the behavior of our formulations on the different LARPs studied. The tests have been run under the same settings as previous computational experiments for MDRPPs.

5.5.1 Set of benchmark instances

The sets of instances used in the computational experiments are adapted from the MDRPPs benchmark instances used in Chapters 3.5.1 and 4.4.1. We have preserved from the original instances the set of required edges and the routing cost function c. The maximum number of facilities to be located has been fixed to $p = 4$. The potential locations for the facilities were chosen randomly from the set of vertices, ensuring that no component had more than one potential location. Potential locations were assigned to components according to some weights p_k, $k \in K$, defined as the sum of a fixed parameter $r = 0.2$, plus a parameter based on the number of required edges, defined as the ratio between the number of required edges in that component and the total number of required edges. That is $p_k = 0.2 + |R_k|/|R|$, for all $k \in K$. For the considered set of benchmark instances the resulting values were always smaller than one, so for each component $k \in K$, p_k was taken as the probability that component k hosted a potential facility location. Then, for each component a number r_k was randomly drawn from a continuous uniform distribution $U[0,1]$, and the component was allocated a potential site when $p_k \leq r_k$. In that case, the vertex of V_k where the potential location was actually located was obtained by randomly generating a number v from a discrete uniform distribution $U[1, |V_k|]$. To generate the set-up costs of the potential locations, for each instance I we have taken from Chapter 4.4.2 the optimal value of the instance solved as an MC-MDRPP with two and four depots, V_I^2 and V_I^4, respectively. Then the value $V_I = |(V_I^2 - V_I^4)|/2$ was taken as the average set-up cost per facility for that instance. Thus, the values f_d, $d \in D$ for instance I have been randomly generated from a discrete uniform distribution $U[V_I/2, 3V_I/2]$. Finally, the capacity of each potential location, b_d,was randomly generated from a discrete uniform distribution $U[|R|/4, 3|R|/4]$. Note that, on average, four open facilities are sufficient to serve all the demand, which is consistent with the selected value of p.

Table 5.2: Characteristics of the instances

	$\|D\|$		$\|D\|$		$\|D\|$		$\|D\|$		$\|D\|$
D16	4–6	G16	4–7	R20	5–8	P	4–6	MAD	4-33
D36	4–10	G36	4–7	R30	5–8	ALB	5	URP5	5–37
D64	5–16	G64	5–15	R40	7–13	ALB2	4–26	URP7	7–43
D100	5–17	G100	5–18	R50	5–17	GRP	5–23		

Table 5.2 shows the number of potential locations ($|D|$). When not all the instances of the group have the same value, the minimum and maximum values are given.

5.5.2 Results for Min-cost p-LARP and Min-cost LARP

Tables 5.3 and 5.4 show, for MC-p-LARP and MC-LARP, respectively, the aggregated results obtained, for each group of instances, with the three-index formulation (3IF), its reinforcement with the optimality condition O8, (3IF-O8), and the two-index formulation (2IF). As before, the columns under $\sharp Opt_0$ and Gap_0 report the number of instances in the group that were optimally solved at the root node and the average percentage gap at the root node with respect to the optimal or best known solution at termination. Similarly, the next two columns under $\sharp Opt$ and Gap give the same information at termination: the number of instances solved to optimality and the average percent gap with respect to the optimal or best known solution. Columns under $Nodes$ represent the average number of nodes explored in the search tree. The columns under CPU give the average of the total computing times in seconds. Finally, the columns under $\sharp D$ show the average number of opened facilities.

Table 5.3: Computational results for the MC-p-LARP

	$3IF$						$3IF-O8$						$2IF$					
	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	9/9	0	-	-	0	0.10	8/9	0.08	9/9	0	0.22	0.11	9/9	0	-	-	0	0.02
D36	3/9	1.62	9/9	0	51.56	17.35	3/9	1.63	9/9	0	31.22	9.81	8/9	0.52	9/9	0	1.22	0.14
D64	0/9	2.50	9/9	0	164.00	519.19	0/9	2.68	9/9	0	108.56	328.62	8/9	0.05	9/9	0	0.56	0.48
D100	0/9	4.00	2/9	2.39	551.56	12361.85	0/9	3.59	4/9	1.31	639.56	11230.97	2/9	0.63	9/9	0	11.22	12.41
G16	6/9	3.01	9/9	0	3.33	0.26	7/9	2.31	9/9	0	2.33	0.21	9/9	0	-	-	0	0.03
G36	3/9	2.31	9/9	0	13.00	10.08	3/9	2.13	9/9	0	4.33	2.87	8/9	0.49	9/9	0	0.33	0.10
G64	2/9	3.69	8/9	0.29	349.33	2031.50	3/9	1.89	9/9	0	113.44	186.33	7/9	0.36	9/9	0	9.11	0.33
G100	1/9	2551	2/9	24.96	24.96	12846.15	1/9	2.06	5/9	1.20	173.56	8931.24	5/9	0.36	9/9	0	2.78	0.95
R20	3/5	3.96	5/5	0	5.80	0.27	4/5	10.38	5/5	0	0.80	0.22	5/5	0	-	-	0	0.05
R30	2/5	2.55	5/5	0	4.80	1.77	3/5	0.13	5/5	0	1.80	1.75	5/5	0	-	-	0	0.08
R40	2/5	1.01	5/5	0	24.20	44.01	3/5	0.78	5/5	0	52.80	54.43	3/5	0.23	5/5	0	1.40	0.30
R50	3/5	1.16	5/5	0	6.40	31.32	4/5	0.40	5/5	0	4.20	35.80	4/5	0.07	5/5	0	0.40	0.21
P	13/24	1.56	24/24	0	19.13	11.19	13/24	1.31	24/24	0	5.38	2.62	17/24	0.31	24/24	0	2.00	0.20
ALB	0/2	2.10	2/2	0	369.50	4739.15	0/2	2.24	2/2	0	70.50	1015.20	2/2	0	-	-	0.50	2.38
ALB2	-	-	-	-	-	-	-	-	-	-	-	-	5/15	1.11	15/15	0	42.87	17.23
GRP	-	-	-	-	-	-	-	-	-	-	-	-	2/10	2.46	10/10	0	276.50	59.83
MAD	-	-	-	-	-	-	-	-	-	-	-	-	8/15	0.35	15/15	0	74.07	251.52
U500	-	-	-	-	-	-	-	-	-	-	-	-	0/7	2.65	3/7	0.58	482.29	60832.93
U700	-	-	-	-	-	-	-	-	-	-	-	-	1/8	18.41	2/8	18.04	56.00	70099.98

Table 5.4: Computational results for the MC-LARP

	$3IF$						$3IF-O8$						$2IF$					
	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	5/9	2.39	9/9	0	3.33	0.16	5/9	1.37	9/9	0	1.22	0.12	9/9	0.00	-	-	0	0.08
D36	3/9	2.13	9/9	0	19.44	6.20	3/9	1.84	9/9	0	17.78	7.96	5/9	0.88	9/9	0	5.44	0.29
D64	0/9	3.02	9/9	0	80.44	180.43	0/9	3.01	9/9	0	45.22	127.89	3/9	0.51	9/9	0	4.44	1.19
D100	0/9	3.91	8/9	0.28	482.56	6610.96	0/9	3.78	8/9	0.23	467.00	5888.12	0/9	1.38	9/9	0	24.22	17.35
G16	8/9	0.79	9/9	0	0.56	0.17	8/9	0.85	9/9	0	0.22	0.12	7/9	0.73	9/9	0	0.78	0.06
G36	9/9	1.94	9/9	0	5.33	2.48	6/9	1.39	9/9	0	4.11	2.33	4/9	1.46	9/9	0	1.89	0.19
G64	3/9	1.78	9/9	0	11.67	28.01	2/9	1.94	9/9	0	11.44	29.86	3/9	0.97	9/9	0	5.11	0.50
G100	0/9	3.61	7/9	0.50	154.22	6303.97	0/9	2.45	8/9	0.10	128.33	4202.78	1/9	1.37	9/9	0	45.56	6.93
R20	4/5	0.14	5/5	0	0.20	0.13	4/5	0.40	5/5	0	0.40	0.23	5/5	0	-	-	0	0.06
R30	4/5	0.21	5/5	0	0.80	1.08	3/5	0.42	5/5	0	0.80	1.15	2/5	0.67	5/5	0	1.00	0.13
R40	3/5	0.43	5/5	0	17.00	33.97	3/5	0.43	5/5	0	18.80	28.68	2/5	1.20	5/5	0	12.60	0.78
R50	3/5	0.67	5/5	0	10.00	55.23	3/5	0.68	5/5	0	12.60	54.80	2/5	3.24	5/5	0	12.40	0.56
P	10/24	2.27	24/24	0	13.25	4.33	9/24	3.26	24/24	0	12.08	5.12	8/24	2.52	24/24	0	11.38	0.43
ALB	0/2	2.45	2/2	0	131.50	1225.94	0/2	2.59	2/2	0	57.00	882.60	1/2	1.12	2/2	0	3.50	2,09
ALB2	-	-	-	-	-	-	-	-	-	-	-	-	4/15	1.66	15/15	0	93.87	36.75
GRP	-	-	-	-	-	-	-	-	-	-	-	-	1/10	2.99	10/10	0	260.70	104,90
MAD	-	-	-	-	-	-	-	-	-	-	-	-	4/15	0.44	15/15	0	92.53	547.17
U500	-	-	-	-	-	-	-	-	-	-	-	-	0/7	1.78	4/7	0.50	449.71	45425.08
U700	-	-	-	-	-	-	-	-	-	-	-	-	0/8	21.43	2/8	21.09	23.50	61583.30

Note that the last five sets corresponding to medium and large instances were solved only with the two-index formulations. Furthermore, for these sets, we also increased the maximum computing time to 24 hours.

Our results show that, both for MC-p-LARP and MC-LARP, the two-index formulation is more efficient and faster than the two three-index formulations. The formulation 2IF allowed us to solve all the small instances within a few minutes, reducing the computing times of 3IF by 98%. In contrast, the three-index formulations 3IF and 3IF-O8 could not find an optimal solution on 18 instances within the limit time of four hours (15 MC-p-LARP instances and three MC-LARP instances). Moreover, with the two-index formulation 2IF we could also solve all medium instances and one third of the large ones. Finally, note that the number of nodes in the search tree is also smaller with 2IF. Comparing Tables 5.3 and 5.4, it can be observed that the results are quite similar regardless of whether the number of facilities to be opened is restricted or set-up costs are included in the objective function.

Comparing the two three-index formulations it is easy to see that 3IF-O8 outperforms 3IF, in terms of the number of instances solved to optimality and, particularly, in terms of computing times. Nevertheless, as mentioned before, the original two-index formulations still outperform the three-index formulations even when these are reinforced with condition O8.

Table 5.5: Computational results for the MC-p-LARP-UD

	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	6/9	7.80	9/9	0	3.33	0.29
D36	1/9	6.77	9/9	0	297.11	127.77
D64	0/9	8.44	6/9	1.98	1630.89	7311.63
D100	0/9	13.87	0/9	12.19	498.89	14400.86
G16	5/9	11.14	9/9	0	15.44	0.62
G36	1/9	7.01	9/9	0	146.78	54.04
G64	0/9	8.26	6/9	4.43	978.67	4295.41
G100	1/9	68.15	1/9	68.15	210.22	13402.86
R20	4/5	1.94	5/5	0	3.00	0.49
R30	2/5	7.86	5/5	0	9.00	1.72
R40	1/5	8.77	5/5	0	493.40	1103.77
R50	0/5	9.35	5/5	0	45.00	111.91
P	9/24	4.12	23/24	0.10	164.25	613.53
ALB	0/2	50.65	1/2	50.00	192.50	7893.08

Tables 5.5 and 5.6 show the results for the models MC-p-LARP-UD and MC-LARP-UD, respectively, which extend the previous models including cardinality constraints on the number of users that can be served from each open facility. As mentioned above, these models were handled with the three-index formulation for being able to reproduce the routes from each open facility once

Table 5.6: Computational results for the MC-LARP-UD

	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	4/9	9.98	9/9	0	9.78	0.46
D36	1/9	6.91	9/9	0	334.89	131.88
D64	0/9	9.65	7/9	2.22	1413.89	6324.78
D100	0/9	12.37	0/9	10.45	499.11	14412.60
G16	5/9	5.32	9/9	0	13.67	0.61
G36	0/9	6.25	9/9	0	77.78	22.44
G64	0/9	10.33	6/9	5.99	1135.67	6562.05
G100	0/9	39.99	1/9	39.34	299.22	13551.45
R20	4/5	2.07	5/5	0	3.00	0.49
R30	1/5	1.77	5/5	0	9.00	1.93
R40	1/5	5.76	5/5	0	749.60	1626.61
R50	0/5	8.19	5/5	0	45.00	146.88
P	7/24	3.97	23/24	0.12	194.79	612.58
ALB	0/2	51.02	1/2	50.00	302.00	7694.61

the values of the decision variables are known. Like in the uncapacitated case, the behavior of models MC-p-LARP-UD and MC-LARP-UD is similar. However, comparing the results with the corresponding version without cardinality constraints we can see, as expected, that the cardinality version is more difficult. This translates into a lower number of instances optimally solved, a larger number of explored nodes, and an increase in the computing time.

5.5.3 Results for Min-max p-LARP

Tables 5.7 and 5.8 report the results obtained with the three-index formulation for the models in which the min-max objective function is considered.

Table 5.7: Computational results for the MM-p-LARP

	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	1/9	36.54	9/9	0	57.22	2.09
D36	0/9	51.44	6/9	5.29	2942.56	7995.30
D64	0/9	58.61	1/9	46.61	1610.78	12985.89
G16	2/9	31.39	9/9	0	34.56	3.90
G36	0/9	41.42	8/9	0.79	804.00	2222.10
G64	0/9	57.60	1/9	46.32	406.78	14400.20
R20	0/5	57.09	5/5	0	246.80	19.69
R30	0/5	53.26	5/5	0	622.40	158.98
R40	0/5	65.05	4/5	10.43	2174.00	7806.01
R50	0/5	73.01	1/5	60.49	284.00	11537.77
P	4/24	28.07	20/24	1.75	1143.00	3074.70

Dealing with this kind of objective is typically difficult. Consequently, the results obtained for these models are the worst ones, with the lowest number of instances optimally solved and the largest computing times. In spite of this, the proposed algorithm found a proven optimal solution for 70% of the tested instances.

Table 5.8: Computational results for the MM-p-LARP-UD

	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	0/9	45.72	9/9	0	103.78	5.10
D36	0/9	49.33	5/9	5.14	3343.00	9091.33
D64	0/9	53.24	1/9	39.57	1025.67	1328.96
G16	1/9	32.15	9/9	0	17.22	2.77
G36	0/9	39.83	7/9	1.59	1405.00	6287.63
G64	0/9	55.52	0/9	42.98	296.00	14400.21
R20	0/5	58.32	5/5	0	253.40	25.01
R30	0/5	54.33	5/5	0	811.20	282.36
R40	0/5	67.10	4/5	14.28	2933.20	9769.81
R50	0/5	81.84	2/5	55.38	343.40	11333.67
P	1/24	34.49	19/24	1.73	1926.17	3284.99

5.5.4 Analysis of the solutions: cross-comparison of models

We close the computational experiments section by analyzing some characteristics of the solutions produced by the different models. The results concerning the number of facilities open in the optimal solutions of the different formulations are summarized in Table 5.9. As could be expected, when the objective takes into account the overall routing costs, models with facility set-up costs (MC-LARP and MC-LARP-UD) produce, in general, solutions with a smaller number of open facilities than the models where the maximum number of open facilities is only limited by the parameter p (MC-p-LARP, MC-LARP). In particular, MC-LARP produces solutions which, on average, have 33% fewer open facilities than MC-p-LARP. This reduction is not so evident for the corresponding models with unit demands and capacity constraints, where MC-LARP-UD produces solutions which, on average, have a around 7% fewer facilities than MC-p-LARP-UD. Similarly, models with unit demands (MC-p-LARP-UD, MC-LARP-UD) produce, in general, optimal solutions with more open facilities than their non-demand counterparts (MC-p-LARP, MC-LARP). On the contrary, it can be observed that unit demand constraints have very little effect on the number of open facilities in the optimal solutions of models with a makespan objective. MM-p-LARP and MM-p-LARP-UD produce solutions with a very similar number of open facilities; there are only five instances out of 98 where the optimal MM-p-LARP-UD solution opens one more facility than the optimal MM-p-LARP solution.

Table 5.9: Average number of open facilities in the optimal solutions of the different models.

	MC-p-LARP	MC-LARP	MC-p-LARP-UD	MC-LARP-UD	MM-p-LARP	MM-p-LARP-UD
D16	3.33	3.11	3.44	3.11	3.78	3.89
D36	2.56	1.56	3.78	3.44	4.00	4.00
D64	2.22	1.44	3.22	3.00	4.00	4.00
D100	3.67	1.89	3.67	3.00		
G16	2.33	1.22	3.11	2.78	3.56	3.56
G36	2.56	1.22	3.56	2.78	4.00	4.00
G64	1.78	1.00	3.56	2.78	4.00	4.00
G100	2.56	1.11	3.00	2.83		
R20	2.00	2.00	2.60	2.60	4.00	4.00
R30	2.60	2.20	3.00	3.20	4.00	4.00
R40	2.80	2.40	3.80	3.60	4.00	4.00
R50	3.20	2.40	3.40	3.40	3.75	3.67
P	3.38	1.13	3.42	2.63	3.58	3.50
ALB	3.50	3.00	2.00	3.00		
Avg.	2.75	1.83	3.25	3.02	3.88	3.87

Since the models with capacity constraints have shown to be notably more difficult to solve than their uncapacitated counterparts we have also investigated how often optimal solutions to models without capacity constraints are feasible (and therefore optimal) for their capacitated versions.

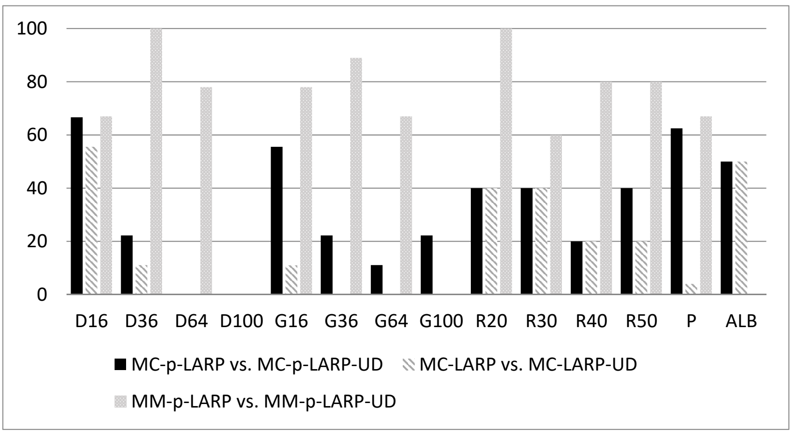


Figure 5.3: Percentage of optimal solutions of uncapacitated models that are feasible for the capacitated counterpart

Figure 5.3 illustrates that the makespan model is clearly more successful in this respect, producing a percentage of feasible solutions for its capacitated counterpart, which ranges in 60–100, depending on the type and size of the instances. In contrast, the capability of producing feasible solutions for their capacitated versions of the models that include the overall routing costs in their objective is quite small, particularly for the more time-consuming instances. It is worth noting that no optimal solution to MC-p-LARP or MC-LARP was feasible for MC-p-LARP-UD and MC-LARP-UD, respectively, with the D64 and the D100 sets of instances.

Finally, we also analyze the *robustness* of the uncapacitated models (MC-p-LARP, MC-LARP, MM-p-LARP), measured in terms of their capability of producing good quality solutions for the other models. For this, the optimal solutions to each model in $\mathcal{F} = \{MC{-}\mathrm{p}{-}LARP, MC - LARP, MM{-}\mathrm{p}{-}LARP\}$ have been evaluated relative to the objectives of the other models, and compared to their optimal values. In particular, let $\overline{x}^i$, denote an optimal solution to formulation $i \in \mathcal{F}$ for a given instance, and $\overline{v}^i$ its optimal value. Let also v^{ij} denote the objective function value of solution $\overline{x}^i$, relative to the objective function of formulation $j \in \mathcal{F}$, $j \neq i$. Table 5.10 gives, for each model $i \in \mathcal{F}$, the averages of the percentages $100(v^{ij} - \overline{v}^i)/\overline{v}^i$, over all the instances of each set of benchmark instances, for each model $j \neq i$.

Table 5.10: Cross-comparison of optimal values to the different models.

	MC-p-LARP		MC-LARP		MM-p-LARP	
	MC-LARP	MM-p-LARP	MC-p-LARP	MM-p-LARP	MC-p-LARP	MC-LARP
D16	1.20	12.63	3.25	20.00	9.64	8.82
D36	2.66	128.20	1.34	178.05	25.95	29.76
D64	1.08	104.65	0.50	138.48	40.08	43.11
D100	2.04		0.56			
G16	9.61	63.89	5.00	187.96	29.26	49.09
G36	8.60	85.40	1.96	168.47	35.53	48.01
G64	3.36	130.23	0.51	159.87	16.39	25.90
G100	3.59		2.17			
R20	1.38	55.06	1.78	55.06	18.10	30.29
R30	2.50	52.10	1.33	93.45	19.31	32.35
R40	1.40	118.75	1.19	138.99	21.68	20.41
R50	1.08	29.49	2.94	51.90	27.56	26.01
P	14.79	36.21	15.23	197.06	10.18	26.23

As can be seen from Table 5.10, the models that include the overall routing costs produce, in general, solutions that are not good for the makespan objective. This is particularly true for MC-LARP, which includes the facilities set-up cost in the objective. The converse holds since the makespan model also produces optimal solutions that, in general, are not of good quality for MC-p-LARP or MC-LARP. On the other hand, not surprisingly, MC-p-LARP produces, in general optimal solutions that are good for MC-LARP, and vice versa. In this sense, the obtained results show a slight superiority of MC-LARP over MC-p-LARP.

Chapter 6

Target Rural Postman Problems

In this chapter we introduce TVARPs on undirected graphs. These problems combine arc routing and linear ordering decisions. Broadly speaking they are single-depot ARPs that consist of finding a tour that starts at the depot, visits all required edges according to some ordering and returns to the depot. The objective aims at balancing two different criteria: the preferences associated with the relative order in which the targets are visited, and the routing cost of the tours that serve all the required edges. In the problems that we study the targets correspond to the components defined by required edges.

We first consider a general TVARP model, referred to as Target-Visitation Rural Postman Problem (TVRPP), with no specific constraints other than visiting all required edges. In the second model that we address, referred to as Clustered Target-Visitation Rural Postman Problem (CTVRPP), it is imposed that all the edges in the same connected component are visited consecutively. In both cases, the goal is to determine a route that serves all required edges taking into account the routing cost of the traversed edges as well as the profit associated with the preferences for the relative position of components in the routes.

For each of the problems, mathematical programming formulations are presented together with families of valid inequalities that reinforce their linear relaxation. Finally, numerical results obtained from branch-and-cut algorithms based on the proposed formulations, are presented and analyzed.

6.1 Formal definition

TVARPs are defined on a undirected complete graph $G = (V, E)$ where V is the vertex set, $|V| = n$, and E is the edge set, $|E| = m$. As in all previous chapters of this thesis, we denote by $R \subset E$ the set of required edges, and by C_k, $k \in K$ the corresponding components. We assume there are at least three clusters and C_1 contains the depot, $d = \{1\} \in V$. Furthermore, there are two types of weights: a non-negative real cost function c defined on the edges of G, and a non-negative real profit a associated with every pair of clusters, $h, k \in K$. In particular, the profit a_{hk} will be collected if the last required edge served in cluster h is visited before the last required edge served in cluster k.

Again 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. In order to formulate the studied models, we transform the undirected graph $G = (V, E)$ to a directed graph $N = (V, A)$, where each edge $e = \{u, v\} \in E$ is replaced by two arcs $a = (u, v)$, $a' = (v, u) \in A$. Hence, for any non-empty vertex subset $S \subset V$, $\delta^+(S) = \{(u, v) \in A | u \in S, v \in V \setminus S\}$ denotes the set of arcs from S to $V \backslash S$ and $\delta^-(S) = \{(u, v) \in A | u \in V \backslash S, v \in S\}$ denotes the set of arcs from $V \backslash S$ to S.

We use the term *net profit* to denote the difference between the total profit for the collected preferences minus the cost of the route.

Definition 6.1.1.

- *The TVRPP is to find a route that serve all the required edges at maximum net profit.*

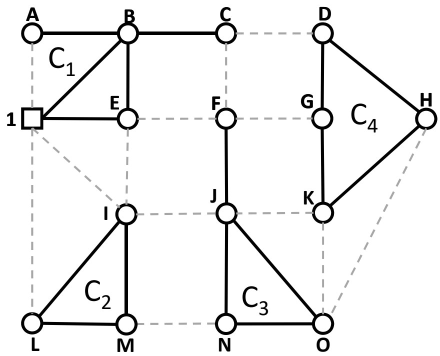
- *The C-TVRPP is to find a route that serve all the required edges at maximum net profit imposing that edges in the same cluster are sequentially visited.*

It is worth specifying when a profit preference is collected. For the C-TVRPP where the required edges of a component are served consecutively, the profit a_{hk} is collected if all required edges of cluster h are served before any required edge of cluster k.

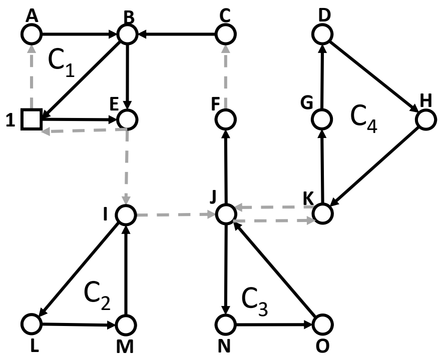
6.1.1 Illustrative example

We next present a small example to illustrate the TVRPP and the C-TVRPP, and to highlight the difference between both models. We consider the graph depicted in Figure 6.1.a with with 16 nodes and four clusters. Solid lines represent the required edges and dotted edges non-required ones. To simplify the picture only a few (dotted lines) are drawn. The routing cost of all edges within a cluster is one, i.e., $c_{ij} = 1$ for all $(i, j) \in E_k, k \in K$. The routing cost of all the edges connecting a given pair of components is the same, i.e. $c_{ij} = C_{hk}$, for all $i \in V_h, j \in V_k, h, k \in K$. These costs are shown in matrix $C = (C_{hk})_{h,k \in K}$ below. The profit matrix $A = (a_{hk})_{h,k \in K}$ is also shown below.

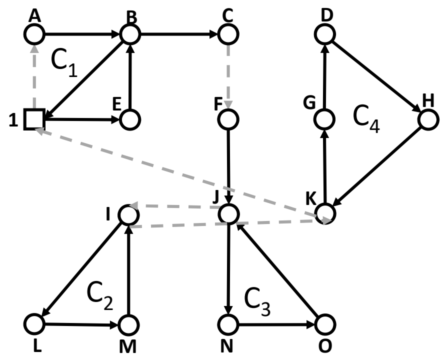
$$C = \begin{pmatrix} - & 20 & 1 & 10 \\ 1 & - & 1 & 5 \\ 1 & 1 & - & 1 \\ 1 & 10 & 10 & - \end{pmatrix} \qquad A = \begin{pmatrix} - & 0 & 0 & 10 \\ 10 & - & 10 & 10 \\ 10 & 0 & - & 0 \\ 0 & 0 & 10 & - \end{pmatrix}$$



(a) Input graph



(b) TVRPP Optimal solution



(c) C-TVRPP Optimal solution

Figure 6.1: Illustrative example of TVARP models.

Figure 6.1.b shows an optimal solution for the TVRPP. If consists of the route: $(1, E)$, (E, I), (I, L), (L, M), (M, I), (I, J), (J, N), (N, O), (O, J), (J, K), (K, G), (G, D), (D, H), (H, K), (K, J), (J, F), (F, C), (C, B), $(B, 1)$, $(1, A)$, (A, B), (B, E), and $(E, 1)$. The cost of the route is 51 units, corresponding to 18 required edges with value 1, plus edges connecting clusters: (E, I), (I, J), (J, K), (K, J), and (F, C) with cost: 20, 1, 1, 10, and 1, respectively. The order in which service to clusters is completed is: C_2–C_4–C_3–$C1$. Thus, the profits that are collected are a_{24}, a_{23}, a_{21}, a_{43}, a_{41}, and a_{31}, with a total profit of 50 units. Thus, the net profit is -1.

Figure 6.1.c shows an optimal solution for the C-TVRPP. Now all required edges in a cluster must be served consecutively. An optimal solution consists of the route: $(1, E)$, (E, B), $(B, 1)$, $(1, A)$, (A, B), (B, C), (C, F), (F, J), (J, N), (N, O), (O, J), (J, I), (I, L), (L, M), (M, I), (I, K), (K, G), (G, D), (D, H), (H, K),

and $(K, 1)$ with a cost of 25 units. The service order of the clusters is now: C_1–C_3–C_2–$C4$, with associate profit of 20. Thus, the corresponding net profit is -5.

Comparing both solutions, it can be seen that, not only the routes are different, but also the sequences of the service order to clusters are different as well.

6.1.2 Complexity and optimality conditions

The TVRPP and the C-TVRRP are NP-hard, since they have as particular cases the RPP and the LOP, which are NP-hard. It is easy to see that the RPP is a particular case of both models when preferences are not considered $(a_{hk} = 0 \; \forall h, k \in K)$. Likewise, it is easy to see that the LOP is also a particular case of both problems when routing cost are not involved $(c_e = 0 \; \forall e \in E)$.

Note that the optimality conditions O1-O5 of the RPP (see Chapter 1.4.1.1) naturally apply to the TVARP, since it is a single vehicle ARP with no capacity constraints. However, a stronger optimality condition can be derived for the C-TVRPP when $G(V, E)$ is a complete input graph. In that case all shortest paths connecting each pair of vertices are represented by some edge of E, so multiple traversals of edges can be hidden by such edges. In particular, for the C-TVRPP where all required edges in the same component are served consecutively, multiple traversals of edges that link connected component can be avoided in optimal solutions.

(O9) **C-TVRPP.** There exists an optimal solution where no edge of T_C is traversed more than once. Since all the required edges in the same cluster are served consecutively only one incoming and one outgoing arc to each cluster will be traversed. Otherwise, if there is a second traversal of a incoming arc to a given cluster, this will be followed by an outgoing arc as, that cluster has already been served. Since the graph is complete, there the second traversal of the incoming arc belongs to some shortest path terminating at a different cluster, which corresponds to some existing edge.

6.2 Mathematical Formulation

We now present linear integer formulations that use binary variables only for the TVARPs we have defined. In particular, we propose two alternative formulations for the TVRPP and one formulation for the clustered version. The main difficulty in these formulations is to compute the visiting order of the clusters, specially when required edges in the same cluster are not necessarily served consecutively.

6.2.1 Formulation for the Target-Visitation RPP

The formulation for the TVRPP uses four sets of binary variables. We denote by $E^y \subset E$ the set of edges that can be traversed twice in an optimal solution, and T the set of periods. The number of periods is upper bounded by the number of possible traversals, $|T| = |E| + |E^y|$. For each $e = \{i, j\} \in E$ and $t \in T$, let x_{ij}^t and x_{ji}^t be binary variables indicating whether or not edge e is traversed for first time in period t, in the direction from i to j and the direction from j to i, respectively. For each $e \in E^y$ and each $t \in T$, let y_{ij}^t and y_{ji}^t be binary variables taking the value one if and only if edge e is traversed for the second time in period t, in the direction from i to j or j to i, respectively. For each $h, k \in K, h \neq k$, let p_{hk} be a binary variable that takes the value one if and only if cluster h is served before cluster k, this means that the service to the last edge of cluster h precedes service to the last edge of cluster k. For each $k \in K$, and $t \in T$, o_k^t indicates if the last required edge of cluster k has been served in period t. Such time period will be referred to as the *completion period* of cluster.

Then, a MILP for the TVRPP is as follows:

$$\text{maximize} \sum_{h\in K}\sum_{k\in K} a_{hk}p_{hk} - \sum_{t\in T}\sum_{e\in E} c_e(x_{ij}^t + x_{ji}^t) - \sum_{t\in T}\sum_{e\in E} c_e(y_{ij}^t + y_{ji}^t) \quad (6.1)$$

subject to

$$\sum_{t\in T}(x_{ij}^t + x_{ji}^t) = 1 \qquad e = \{i,j\} \in R \quad (6.2)$$

$$\sum_{t\in T}(x_{ij}^t + x_{ji}^t) \leq 1 \qquad e = \{i,j\} \in E \setminus R \quad (6.3)$$

$$\sum_{t\in T}(y_{ij}^t + y_{ji}^t) \leq 1 \qquad e = \{i,j\} \in E_y \quad (6.4)$$

$$\sum_{t\in T}(x_{ij}^t + y_{ij}^t) \leq 1 \qquad e = \{i,j\} \in E_y \quad (6.5)$$

$$\sum_{t\in T}(x_{ij}^t + y_{ij}^t)(\delta^+(S)) \geq 1 \qquad S \subseteq V \setminus \{1\} \quad (6.6)$$

$$(x_{ji}^t + y_{ji}^t)(\delta^-(i)) = (x_{ij}^{t+1} + y_{ij}^{t+1})(\delta^+(i)) \quad t = 1, \ldots, |T|-1,\ i \in V \setminus \{1\} \quad (6.7)$$

$$y_{ij}^t \leq \sum_{t'<t} x_{ji}^{t'} \qquad e = \{i,j\} \in E_y,\ t \in T \quad (6.8)$$

$$\sum_{j\in V} x_{1j}^1 = 1 \quad (6.9)$$

$$(x_{ij}^t + x_{ji}^t + y_{ij}^t + y_{ji}^t)(E) \leq 1 \qquad t \in T \quad (6.10)$$

$$p_{hk} + p_{kh} = 1 \qquad h, k \in K,\ h \neq k \quad (6.11)$$

$$p_{hk} + p_{kl} + p_{lh} \leq 2 \qquad h, k, l \in K,\ h \neq k \neq l \quad (6.12)$$

$$\sum_{t\in T} o_k^t = 1 \qquad k \in K \quad (6.13)$$

$$\sum_{k\in K} o_k^t \leq 1 \qquad t \in T \tag{6.14}$$

$$(x_{ij}^t + x_{ji}^t)(R_k) \leq \sum_{t'\geq t} o_k^{t'} \qquad k \in K,\ t \in T \tag{6.15}$$

$$o_k^t \leq (x_{ij}^t + x_{ji}^t)(R_k) \qquad k \in K,\ t \in T \tag{6.16}$$

$$p_{hk} + o_h^t \leq \sum_{p=t+2}^{|T|} o_k^p + 1 \qquad h, k \in K,\ h \neq k,\ t = 1, \ldots, |T|-1 \tag{6.17}$$

$$x_{ij}^t, x_{ji}^t \in \{0,1\} \qquad e = \{i,j\} \in E,\ t \in T \tag{6.18}$$

$$y_{ij}^t, y_{ji}^t \in \{0,1\} \qquad e = \{i,j\} \in E_y,\ t \in T \tag{6.19}$$

$$p_{hk} \in \{0,1\} \qquad h, k \in K,\ h \neq k \tag{6.20}$$

$$o_k^t \in \{0,1\} \qquad k \in K,\ t \in T \tag{6.21}$$

Equalities (6.2) guarantee that all the required edges are served in some period in one of the two possible directions. Inequalities (6.3) ensure that edges belonging to $E \setminus R$ are traversed at most in one direction and period. Constraints (6.4) impose that at most one direction can be used for the second traversal of an edge, whereas (6.5) force that an edge is not traversed twice in the same direction. Inequalities (6.6) are the well-known connectivity constraints. They impose that, at any time period, at least one edge crosses the cut-set $\delta(S)$ from S to its complement. Constraints (6.7) ensure that in each vertex the number of incoming edges in period t is equal to the number of outgoing edges in period $t-1$. These constraints also guarantee the parity of every vertex. Inequalities (6.8) impose that an edge cannot be traversed for a second time unless that it has been traversed for a first time before in the opposite direction. Equality (6.9) forces that the edge traversed in the first period is incident with the depot. Constraints (6.10) guarantee that in each period only one edge is traversed. Inequalities (6.11) are LOP constraints stating that for any pair of clusters one must precede the other one. Dicycle constraints (6.12) model the fact that it is not possible to order h before k, k before l, and l before h. Inequalities (6.13) guarantee that each cluster serves is last required edge in only one period. Constraints (6.14) force that in each period service is completed for, at most, one cluster. Constraints (6.15) and (6.16) give the relation between variables x and o. Inequalities (6.15) force that the period in which the completion period of a cluster should be greater or equal than the period in which each of the edges of that cluster is served. Moreover, (6.16) impose that completion period of a cluster is a period when some required edge of that cluster is served. Constraints (6.17) model the relation between variables p and o: if completion period of cluster h is period t, and cluster h is served before cluster k, then completion period of cluster k must be in a period after t. Binary conditions of the variables x, y, p, and o derived from their definition are reflected in constraints (6.18)–(6.21).

The above formulation contains $|2(E \times T)|$ x variables, $|2(E^y \times T)|$ y variables, $|K \times (K-1)|$ p variables, and $|K \times T|$ o variables. There are $|R|$ equalities of type (6.2), $|E \setminus R|$ constraints (6.3), $|E^y|$ inequalities of types (6.4) and (6.5), $|(T-1) \times (V-1)|$ constraints (6.7), $|E^y \times T|$ inequalities (6.8), $|T|$ constraints of types (6.10), (6.14) and (6.16), $|K \times (K-1)|$ inequalities (6.11), $|K \times (K-1) \times (K-2)|$ constraints of type (6.12), $|K|$ inequalities (6.13), $|K \times T|$ inequalities (6.15), and $|K \times (K-1) \times (T-1)|$ constrains of type (6.17). The size of the family constraint (6.6) is exponential in $|V|$.

6.2.1.1 Variable elimination

The nature of the TVRPP allows us to eliminate, a priori, the following variables:

- Since any feasible route starts at the depot, and we assume that the depot is located at vertex $i = 1$, for the first period $t = 1$, only variables x^1_{1j}, $j \in V$ are defined. Hence, all variables x^1_{ij}, with $i \neq 1$ can be eliminated. Furthermore, no variable y^1_{ij} is defined.
- The minimum number of periods needed to complete the service of cluster $k \in K$ is R_k. Hence, for cluster 1, we eliminate all variables o^t_1 with $t < R_1$, and for all other clusters we eliminate all variables o^t_k with $t < 1 + R_k$.
- After completing the graph with shortest paths, we can assume that no optimal solution will have two consecutive *traversals* without service, since they can be concatenated into a single one. Hence, the maximum number of edges in an optimal solution is bounded by above by $\sum_{k \in K} |R_k| + \frac{n_{odd}}{2} + 2|K| - 1$, where n_{odd} denotes the number of vertices with odd R-degree. In practice, this is smaller that the previous bound $T = |E| + |E^y|$, so it allows to reduce the number of variables associated with T.

6.2.1.2 Valid inequalities

Below we introduce some families of valid inequalities that can be used to reinforce the LP relaxation of the formulation presented above.

- If the completion period of cluster h is smaller that the completion period of cluster k, then there is an outgoing edge in any subset $S \subset V \setminus V_k,\ V_h \subset S$:

$$p_{hk} \leq \sum_t x^t(\delta^+(S)) \quad S \subset V \setminus V_k,\ V_h \subset S. \tag{6.22}$$

- If the completion period of cluster h is t, then there is an outgoing edge from cluster h in the next period, $t+1$:

$$o^t_h \leq x^{t+1}(\delta^+(V_h)) \quad h \in K,\ t = 1, \ldots, |T| - 1. \tag{6.23}$$

- If the completion period of cluster h is t, then no incoming edge to cluster h can be traversed in periods t or $t+1$:

$$o_h^t + x^t(\delta^-(V_h)) + x^{t+1}(\delta^-(V_h)) \leq 1 \quad h \in K,\ t = 1, \dots, |T|-1 \tag{6.24}$$

- The number of periods between the completion periods of any pair of clusters, h and k, must be at least one, corresponding with the traversal of an edge belonging the cut-set. So, if the completion period of cluster h is t, then the completion period of cluster k cannot be $t+1$, and vice versa:

$$o_h^t + o_h^{t+1} + o_k^t + o_k^{t+1} \leq 1 \quad h, k \in K,\ h \neq k,\ t = 1, \dots, |T|-1 \tag{6.25}$$

- No two edges in the cut-set of some cluster can be traversed in consecutive periods:

$$\sum_{k \in K} \left(x^t(\delta^-(V_k) + x^{t+1}(\delta^-(V_k)) \right) \leq 1 \quad t = 1, \dots, |T|-1 \tag{6.26}$$

- A *connection* of two clusters, h and k, in period $t \in T$ must be preceded by a service of required edge in cluster h in $t-1$ and followed by another *service* of required edge in cluster k in period $t+1$:

$$x^t(\delta^-(V_h : V_k)) \leq (x_{ij}^{t-1} + x_{ji}^{t-1})(R_h) \quad k \in K,\ t = 2, \dots, |T|-1 \tag{6.27}$$

$$x^t(\delta^-(V_h : V_k)) \leq (x_{ij}^{t+1} + x_{ji}^{t+1})(R_k) \quad k \in K,\ t = 2, \dots, |T|-1 \tag{6.28}$$

- Inequalities (6.8) can be reinforced for second traversals of edges connecting two clusters (i.e. in the MST of G_C), by taking into account that no optimal solution will use two connecting edges in two consecutive periods:

$$y_{ij}^t \leq \sum_{t' < t-1} x_{ji}^{t'} \quad e = \{i, j\} \in E_y \setminus R,\ t \in T \tag{6.29}$$

6.2.1.3 Alternative definition of *termination* variables

Below we present an alternative formulation for the TVRPP where the time completion variables o associated with clusters are redefined. For each cluster $k \in K$ and $t \in \tilde{T} = T \cup \{|T|+1\}$ let binary variable $\tilde{o}_k^t$ take the value one if and only if the completion period of cluster k is at least t. Using these variables we can reformulate the TVRPP using the same meaning for the remaining variables, x, y and p as:

$$\text{maximize} \sum_{h \in K} \sum_{k \in K} a_{hk} p_{hk} - \sum_{t \in T} \sum_{e \in E} c_e(x_{ij}^t + x_{ji}^t) - \sum_{t \in T} \sum_{e \in E} c_e(y_{ij}^t + y_{ji}^t) \tag{6.1}$$

subject to

$$\sum_{t\in T}(x_{ij}^t + x_{ji}^t) = 1 \qquad e=\{i,j\}\in R \tag{6.2}$$

$$\sum_{t\in T}(x_{ij}^t + x_{ji}^t) \le 1 \qquad e=\{i,j\}\in E\setminus R \tag{6.3}$$

$$\sum_{t\in T}(y_{ij}^t + y_{ji}^t) \le 1 \qquad e=\{i,j\}\in E_y \tag{6.4}$$

$$\sum_{t\in T}(x_{ij}^t + y_{ij}^t) \le 1 \qquad e=\{i,j\}\in E_y \tag{6.5}$$

$$\sum_{t\in T}(x_{ij}^t + y_{ij}^t)(\delta^+(S)) \ge 1 \qquad S\subseteq V\setminus\{1\} \tag{6.6}$$

$$(x_{ji}^t + y_{ji}^t)(\delta^-(i)) = (x_{ij}^{t+1} + y_{ij}^{t+1})(\delta^+(i)) \quad t=1,\ldots,|T|-1,\ i\in V\setminus\{1\} \tag{6.7}$$

$$y_{ij}^t \le \sum_{t'<t} x_{ji}^{t'} \qquad e=\{i,j\}\in E_y,\ t\in T \tag{6.8}$$

$$\sum_{j\in V} x_{1j}^1 = 1 \tag{6.9}$$

$$(x_{ij}^t + x_{ji}^t + y_{ij}^t + y_{ji}^t)(E) \le 1 \qquad t\in T \tag{6.10}$$

$$p_{hk} + p_{kh} = 1 \qquad h,k\in K,\ h\ne k \tag{6.11}$$

$$p_{hk} + p_{kl} + p_{lh} \le 2 \qquad h,k,l\in K,\ h\ne k\ne l \tag{6.12}$$

$$(x^t + x^t)(E_k) + (y^t + y^t)(R_k\cap E_y) \le \tilde{o}_k^t \qquad k\in K,\ t\in T \tag{6.30}$$

$$\tilde{o}_k^t - \tilde{o}_k^{t+1} \le x_{ij}^t(R_k) \qquad k\in K,\ t\in T \tag{6.31}$$

$$p_{hk} + \tilde{o}_h^t \le \tilde{o}_k^{t+2} + 1 \qquad h,k\in K,\ h\ne k,\ t=1,\ldots,|T| \tag{6.32}$$

$$\tilde{o}_1^t = 1 \qquad t\le |R_1| \tag{6.33}$$

$$\tilde{o}_k^t = 1 \qquad t\le |R_k|+1,\ k\in K\setminus\{1\} \tag{6.34}$$

$$\tilde{o}_k^{|T|+1} = 0 \qquad k\in K \tag{6.35}$$

$$\tilde{o}_k^{t+1} \le \bar{o}_k^t \qquad k\in K,\ t\in T \tag{6.36}$$

$$x_{ij}^t, x_{ji}^t \in \{0,1\} \qquad e=\{i,j\}\in E,\ t\in T \tag{6.18}$$

$$y_{ij}^t, y_{ji}^t \in \{0,1\} \qquad e=\{i,j\}\in E_y,\ t\in T \tag{6.19}$$

$$p_{hk} \in \{0,1\} \qquad h,k\in K,\ h\ne k \tag{6.20}$$

$$\tilde{o}_k^t \in \{0,1\} \qquad k\in K,\ t\in \tilde{T} \tag{6.37}$$

Inequalities (6.2)–(6.12), referring to the design of the route and the preferences, as well as the domain of the variables x, y, and p (6.18)–(6.20) are exactly the same as in the previous formulation. Constraints (6.30) and (6.31) give the relation between variables x and the new variables $\tilde{o}$. Inequalities (6.30) force that if there some required edge of cluster k is traversed at period t, then the completion period of cluster k is at least t, whereas, (6.16) impose that the completion period of a cluster corresponds to a period when one of its required edges is served. Inequalities (6.32) model the relation between variables p and $\tilde{o}$: if the the completion period of cluster h is t, and cluster h is served before cluster k, then the completion period of cluster k is $t+2$ or later. Constraints (6.33)–(6.36) are derived from the nature of variables $\tilde{o}$. Equalities (6.33) and (6.34) model the fact that at least $|R_k|$ periods are needed to finish the service of cluster k, when it does not contain the depot. Equalities (6.35) ensure that all clusters have been served by period $T+1$. Inequalities (6.36) guarantee that if the completion period of a cluster is at least $t+1$, then its completion period is at least t.

Some valid inequalities can be used to reinforce the LP relaxation of the formulation (6.1)–(6.37).

- (6.30) can be reinforced by including the incoming edges in the cut-set of $\delta^-(V_k)$:

$$(x^t + y^t)(\delta^-(V_k)) + x^t(E_h) + y^t(E_k \cap E_y) \le \tilde{o}_k^t \quad k \in K,\ t \in T \tag{6.38}$$

- At each period at most one cluster will be *terminated*:

$$\sum_{k\in K} (\tilde{o}_k^t - \tilde{o}_k^{t+1}) \le 1 \quad t \in T \tag{6.39}$$

6.2.2 Formulation for the Clustered Target-Visitation RRP

In order to formulate the C-TVRPP, we can exploit the fact that all required edges in the same component must be served consecutively. This allows us not to use the completion variables o and also not to use the time index t in the routing variables. Thus, the ILP formulation for the C-TVRRP we give below uses two sets of binary variables. For each arc $(i,j) \in A$, let x_{ij} be a binary variable indicating whether or not arc (i,j) is traversed. For each pair of components or clusters $h, k \in K$ let p_{hk} be a binary variable that takes the value one if and only if cluster h is served before cluster k.

The C-TVRRP formulation is as follows:

$$\text{maximize} \sum_{h,k\in K: h\neq k} a_{hk}p_{hk} - \sum_{\{i,j\}\in E} c_{ij}x_{ij} \tag{6.40}$$

subject to

$$x_{ij} + x_{ji} \ge 1 \qquad \{i,j\} \in R \tag{6.41}$$
$$x_{ij} + x_{ji} \le 1 \qquad \{i,j\} \in E \setminus R \tag{6.42}$$
$$x(\delta^+(S)) \ge 1 \qquad S \subseteq V \setminus \{1\} \tag{6.43}$$
$$x(\delta^+(i)) = x(\delta^-(i)) \qquad i \in V \tag{6.44}$$
$$x(\delta^+(V_k)) \le 1 \qquad k \in K \tag{6.45}$$
$$p_{1k} = 1 \qquad k \in K \setminus \{1\} \tag{6.46}$$
$$p_{hk} + p_{kh} = 1 \qquad h, k \in K \tag{6.47}$$

$$p_{hk} + p_{kl} + p_{lh} \le 2 \qquad h, k, l \in K \tag{6.48}$$

$$x(\delta^+(V_h : V_k)) \le p_{hk} \qquad h, k \in K \setminus \{1\} \tag{6.49}$$

$$x(\delta^+(V_h : V_1)) \le p_{kh} \qquad h, k \in K \; 1 \ne h \ne k \tag{6.50}$$

$$x_{ij}, x_{ji} \in \{0, 1\} \qquad e = \{i, j\} \in E \tag{6.51}$$

$$p_{hk} \in \{0, 1\} \qquad h, k \in K \tag{6.52}$$

Inequalities (6.41) ensure that all required edges are served at least once in some direction. Constraints (6.42) guarantee that no non-required edge is traversed more than once. Recall, that the edges of T_C will be traversed at most once according to the optimality condition O9. Inequalities (6.43) are the connectivity inequalities, which impose that at least one arc must cross the cut-set $\delta^+(S)$. Inequalities (6.44) ensure the parity of every vertex, through the balancing of incoming and outgoing arcs. Constraints (6.45) force that there is only one outgoing arc from each cluster. Equalities (6.46) force that cluster 1, where the depot is located, is served before any other cluster. Constraints (6.47) guarantee that either cluster h precedes cluster k or vice versa. Constraints (6.48) are the dicycle inequalities (6.12). Inequalities (6.49) impose that if there is some arc in the cut-set from cluster h to cluster k, then h is served before k. Binary conditions of the variables x and p derived from their definition are reflected in constraints (6.51) and (6.52).

6.2.2.1 Valid inequalities

Below we introduce some families of valid inequalities to reinforce the LP relaxation of the formulation (6.40)–(6.52).

- If cluster h is served before k, then there is an outgoing arc in any subset $S \subset V \setminus V_k, V_h \subset S$:

$$p_{hk} \le x(\delta^+(S)) \qquad S \subset V \setminus V_k, V_h \subset S \tag{6.53}$$

- If service to cluster h is completed immediately before cluster k, and service to cluster k is completed before service to cluster l, then service to cluster h is completed before service to cluster l:

$$x(\delta^+(V_h : V_k)) + p_{kl} \le 1 + p_{hl} \qquad h, k, l \in K \tag{6.54}$$

6.3 Branch-and-cut for TVARPs

In this section, we present a branch-and-cut algorithms for the TVARPs based on each of the formulations above. In each case, the families of connectivity constraints (6.6) and (6.43), which have exponential size, are initially relaxed and, at each iteration, inequalities violated by the current LP solution are separated and reincorporated.

In each case, the solution algorithm starts with the integrality conditions relaxed, a subset of connectivity constraints, and the incorporation of some valid inequalities to reinforce the solution. In particular, we include connectivity inequalities associated with the subsets defined by the vertices of each cluster, and the valid inequalities (6.25), (6.26), (6.29), and (6.39) for the TVRPP, and (6.54)for the C-TVRPP.

For the TVRPP, the separation problem for the connectivity constraints (6.6) can be solved similarly to the connectivity constraints of the RPP (see Chapter 1.4.1.3). Thus, we first check the connected components in the graph induced by the solution. If there is more than one, each connected component C does not contain the depot, $V(C) \subseteq V \setminus \{d\}$, defines a violated constraint (6.6). Otherwise, we build the tree of min-cuts T relative to capacities given by $\sum_{t \in T} (\bar{x}_{ij}^t + \bar{y}_{ij}^t)$. Then, if the value of a min-cut, $\delta(S)$, is smaller than 1, the inequalities (6.6) associated with the subset S is violated by the current solution.

The above procedure can be adapted to the case of the connectivity constraints of the C-TVRPP, (6.43). Now the tree of min-cuts is computed relative to the capacities vector given by $\bar{x}_{ij}$.

6.4 Computational experience

In this section we present the results of the computational experiments in order to evaluate the performance of the branch-and-cut algorithms. The tests have been run under the same settings as the computational experiments described in the previous chapters. The maximum computing time has been set to four hours. Moreover, the branching rule has been modified to prioritize the variables p associated with precedences information.

6.4.1 Set of benchmark instances

The algorithms were tested on some of the smallest benchmark instances used in the previous Chapters. The set of required edges, the routing cost function c, and the depot $d = 1$ are preserved from the original instances. To define the profit function a_{hk} associated with every pair of clusters $h, k \in K$ we have considered the prize-collecting instance associated with each original RPP instance (Aráoz et al., 2009b) and computed for each cluster $k \in K$ the overall profit of its required edges, P_k. Then, the profit a_{hk} associated with the pair of clusters $h, k \in K$, takes value 1 if $P_h \geq P_k$, and 0 otherwise.

6.4.2 Results for Target-Visitation RPP

Table 6.1 shows the results obtained for the TVRPP with the two alternative formulations, depending the definition of variables o. As before, $\sharp Opt_0$ and

Gap_0 report the number of instances in the group that were optimally solved at the root node and the average percentage gap at the root node with respect to the optimal or best-known solution at termination. Similarly, $\sharp Opt$ and Gap give the same information at termination. Column under $Nodes$ shows the average number of nodes explored in the search tree, and column under CPU gives average total computing times in seconds.

Table 6.1: Computational results for the TVRPP

	$TVRPP$						$TVRPP$:2					
	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	1/9	0.05	7/9	0.01	5939,78	3723,53	1/9	0.04	8/9	0.00	4957.22	3063.94
D36	0/9	46.53	0/9	46.53	1128.44	14432.68	0/9	56.57	2/9	56.57	1191.11	13268.15
G16	3/9	0.00	9/9	0	691.44	311.44	2/9	0.00	9/9	0	715.67	360.46
G36	0/9	39.82	0/9	39.82	830.00	14523.89	0/9	49.69	0/9	49.69	952.56	14434.11
R20	1/5	0.99	5/5	0	255.40	11.69	2/5	1.16	5/5	0	165.20	9.53
R30	0/5	1.91	4/5	0.05	3959.60	3629.12	0/5	1.75	4/5	0.41	2816.00	3414.49
R40	0/5	3.81	1/5	3.51	2242.60	11569.71	0/5	4.71	2/5	4.33	10029.20	9200.81
R50	0/5	2.47	0/5	2.42	868.00	14411.59	0/5	29.42	0/5	29.23	911.20	14407.51
P	1/24	48.98	7/24	48.98	3389.63	10952.41	2/24	37.50	7/24	37.50	2394.29	10786.87

Our results highlight the difficulty of the studied problem. The number of instances solved to optimally with each formulation is low, 33 and 37 out of the set of 80 benchmark instances, respectively. In general, solved instances are small ones with less than 30 vertices. Furthermore, no feasible integer solution was found within the time limit for 13 and 18 instances, respectively.

Note that the obtained results do not allow to conclude that any of the two formulations outperforms the other one. On the contrary, both formulations present similar results in terms of the number of solved instances, the values of the gaps, and the computing times.

6.4.3 Results for Clustered Target-Visitation RPP

Table 6.2 shows the results for the C-TVRPP. The obtained results indicate that the restriction that required edges in the same component must be served consecutively, leads to a easier problem than the non-clustered version. In particular, 96 instances out of the set of 118 benchmark instances are optimality solved, 50 of which at the root node. It is worth mentioning, that the percentage optimality gap of the unsolved instances is nearly null.

Table 6.2: Computational results for the C-TVRPP

	$\sharp Opt_0$	Gap_0	$\sharp Opt$	Gap	$Nodes$	CPU(s)
D16	8/9	0.00	9/9	0	3.78	0.08
D36	3/9	0.00	9/9	0	357.67	17.91
D64	1/9	0.01	6/9	0.00	9001.11	4914.55
D100	2/9	0.00	4/9	0.00	2642.56	8476.48
G16	8/9	9.44	9/9	0	0.33	0.06
G36	3/9	0.00	9/9	0	1191.67	59.68
G64	1/9	0.00	6/9	0.00	8460.11	4875.06
G100	3/9	0.00	4/9	0.00	2719.67	8579.38
R20	5/5	0	5/5	0	0	0.03
R30	4/5	0.06	5/5	0	3.00	0.07
R40	3/5	0.02	5/5	0	10.00	0.14
R50	3/5	0.02	5/5	0	22.00	0.70
P	6/24	37.66	20/24	0.00	14692.63	2940.28
ALB	0/2	0.01	0/2	0.01	4078.50	14401.57

Chapter 7

Conclusions

Some of the most relevant decisions to be taken at the operational level in the management of logistic systems are related to the design of efficient service routes. In this thesis, we have studied three optimization problems addressing this kind of decisions when the demand for service is located at the links of a given network. In particular, we have studied several families of arc routing problems, namely MDRPPs, LARPs, TVARPs.

The first group of problems analyzed in this thesis are MDRPPs, which extend the RPP to the case of several depots. In particular, some properties and dominance relations have been studied for two MDRPP variants. In one model the objective is to minimize the overall routing costs, whereas the second model uses a min-max objective function aiming at minimizing the makespan, the length of the longest route. A worst-case analysis of the Min-cost 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.

Integer linear programming formulations, with three-index variables associated with the traversed edges and the depots where the routes start and end, have been presented for both models where, as usual, the families of constraints that enforce connectivity and parity of solutions are of exponential size. Moreover, an aggregate formulation containing only binary variables has been proposed for the Min-cost MDRPP, in which the variables are associated only to the traversed edges. This alternative formulation includes a new family of inequalities of exponential size that ensure that routes start and end at the same depot. Furthermore, the properties of the polyhedron associated with the compact formulation have been studied.

A branch-and-cut algorithm has been developed for each proposed formulation, where the separation problems of constraints of exponential size are solved heuristically and exactly. The performance of the algorithm has been tested for the two proposed models. For this, in each case we have solved two sets of instances, with two and four depots, respectively. The computational experience for the min-cost objective, shows the superiority of the two-index formulation in terms of efficiency and speed with respect to the three-index formulation. The disadvantage of the disaggregate formulation lies in the increase on the number of variables with the instance size. In particular, for the three-index formulation, 35% and 51% of two- and four-depot instances with up 100 vertices were optimally solved at the root node and these percentages raise to 97% and 92% of instances optimally solved at termination. The good behavior of the compact formulation allowed us to solve larger instances involving up to four depots, 744 vertices, 140 required components and 1000 required edges within reasonable computing times. Nearly 60% of the larger instances were solved at the root node and this percentage increases to 96.4% at termination. The numerical experiments when the objective is to minimize the makespan, indicate that, computationally, this model becomes notably more demanding. Nevertheless, the formulation for this min-max version is indeed successful in producing balanced routes.

The second family of problems studied in this thesis are LARPs, in which location and routing decisions are combined. We have modeled and solved six LARPs with different characteristics. The models differ from each other in their objective function, on whether the number of facilities to be located is upper bounded, or on whether the facilities are capacitated. We have considered min-cost objectives aiming at minimizing the overall routing costs (possibly incorporating facilities set-up costs as well), and min-max objectives aiming at minimizing the makespan. Some of the studied models assume that there are no capacity limitations, whereas other models include a cardinality constraint on the number of users that can be served from an open facility.

Three-index variable formulations have been presented for all the models. The polyhedral analysis carried out for the three-index formulation of the uncapacitated models proves that the main families of constraints are facet defining. Moreover, a two-index variable formulation was also introduced for the min-cost models without capacity constraints, which incorporates a new set of constraints forcing the routes return to their departing facility. All the formulations exploit optimality conditions, which allow using binary decision variables only.

Exact and heuristic separation procedures have been studied for the large-size families of inequalities and exact branch-and-cut solution algorithms have been implemented for the solution of the proposed formulations. Our numerical results demonstrate the good behavior of the algorithms, which were tested on several sets of benchmark instances. For the uncapaciteted min-cost models, where comparisons are possible, our results show the superiority of the two-index formulation in terms of efficiency and speed with respect to the three-index formulations. Thereby with the compact formulations, all instances involving up to 200 vertices, as well as most instances with up to 744 vertices, were solved to optimality, while with the disaggregate one, not all instances involving up to 100 vertices were optimally solved. Despite the difficulty of the models with a makespan objective or with capacity constraints, instances with up to 100 vertices were optimally solved for the makespan objective and for the capacitated versions of the min-cost models.

The last group of studied problems in this thesis are TVARPs, which combine arc routing decisions with linear ordering preferences for the order in which clusters are served. We have modeled and solved two alternative TVARPs: the general case and a clustered version, where it is imposed that all the edges in the same component are served sequentially.

Alternative formulations have been proposed in which all variables are binary. Two formulations model the general case, and use alternative variables to identify the period when service of each cluster has been completed. The third formulation models the clustered version of the target visitation problem.

Branch-and-cut algorithm has been developed for each proposed formulation. The obtained results highlight the difficulty of these models, especially in the general case. For that case, none of the proposed formulations shows relevant differences in terms of the number of solved instances, the values of the percentage optimality gaps, or the computing times. Still, the percentage of instances solved to optimality for the general case and the clustered version is 44% of instances and 81%, respectively.

Bibliography

D. Ahr. Contributions to multiple postmen problems. Technical report, PhD dissertation, Departament of Computer Science, Heidelberg University, Heidelberg, Germany, 2004.

M. Albareda-Sambola. Location-routing and location-arc routing. *In Location Science, Laporte, G. and Nickel, S. and Saldanha da Gama, F., eds. Springer, Berlin Heidelberg*, pages 399–418, 2015.

A. Amaya, A. Langevin, and M. Trépanier. The capacitated arc routing problem with refill points. *Operations Research Letters*, 35(1):45–3, 2007.

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.

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

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

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

C. Arbib, M. Servilio, C. Archetti, and M.G. Speranza. The directed profitable location rural postman problem. *European Journal of Operational Research*, 236(3):811–819, 2014.

A. Arulselvan, C.W. Commander, and P.M. Pardalos. A hybrid genetic algorithm for the target visitation problem. *Naval Research Logistics*, 2007.

R. Baldacci and V. Maniezzo. Exact methods based on node-routing formulations for undirected arc-routing problems. *Networks*, 47(1):52–60, 2006.

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

J.M. Belenguer and E. Benavent. The capacitated arc routing problem: Valid inequalities and facets. *Computational Optimization and Applications*, 10(2):165–187, 1998.

E. Benavent, A. Corberán, I. Plana, and J.M. Sanchis. Min-max k-vehicles windy rural postman problem. *Networks*, 54:216–226, 2009.

E. Benavent, A. Corberán, and J.M. Sanchis. A metaheuristic for the min-max windy rural postman problem with k-vehicles. *Computationqal Management Science*, 7:269–287, 2010.

E. Benavent, A. Corberán, I. Plana, and J.M. Sanchis. New facets and enhanced branch-and-cut for the min-max k-vehicles windy rural postman problem. *Networks*, 58:255–272, 2011.

E. Benavent, A. Corberán, G. Desaulniers, F. Lessard, I. Plana, and J.M. Sanchis. A branch-and-cut for the maximum benefit chinese postman problem. *Networks*, 63: 34–45, 2013.

E. Benavent, A. Corberán, I. Plana, and J.M. Sanchis. Arc routing problems with min-max objectives. In A Corberán and G Laporte, editors, *Arc Routing: Problems, Methods and Applications*, pages 255–280. MOS-SIAM Series on Optimization, Philadelphia, 2014.

Z. Blázsik, T. Bartók, B. Imreh, C. Imreh, and Z. Kovacs. Heuristics on a common generalization of tsp and lop. 17, 01 2006.

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

E.A. Cabral, M.l Gendreau, G. Ghiani, and G. Laporte. Solving the hierarchical chinese postman problem as a rural postman problem. *European Journal of Operational Research*, 155(1):44 – 50, 2004.

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.

M. Colombi, Á. Corberán, R. Mansini, I. Plana, and J.M. Sanchis. The hierarchical mixed rural postman problem. *Transportation Science*, 51(2):755–770, 2016.

M. Colombi, Á. Corberán, R. Mansini, I. Plana, and J.M. Sanchis. The hierarchical mixed rural postman problem: Polyhedral analysis and a branch-and-cut algorithm. *European Journal of Operational Research*, 257(1):1–12, 2017.

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

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

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.

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

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.

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.

M. Drexl, M. Schneider, and D.B. Scheneker. A survey of location-routing problems. Technical report, Darmstadt Technical University, Department of Business Administration, Economics and Law, Institute for Business Studies (BWL), 2013.

M. Dror. *Arc routing: theory, solutions, and applications*. Springer, 2000.

M. Dror, H. Stern, and P. Trudeau. Postman tour on a graph with precedence relation on arcs. *Networks*, 17(3):283–294, 1987.

H.A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems, part i: The chinese postman problem. *Operations Research*, 43(2):231–242, 1995a.

H.A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems, part ii: The rural postman problem. *Operations Research*, 43(3):399–414, 1995b.

E. Fernández and J. Rodríguez-Pereira. Multi-depot rural postman problems. *TOP*, 25(2):340–372, 2017.

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

E. Fernández, G. Laporte, and J. Rodríguez-Pereira. A branch-and-cut algorithm for the multi-depot rural postman problem. *Transportation Science*, (forcoming), 2018.

G.N. Frederickson, M.S. Hecht, and C.E. Kim. Approximation algorithms for some routing problems. In *Foundations of Computer Science, 1976., 17th Annual Symposium on*, pages 216–227. IEEE, 1976.

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 Operation al Research*, 23:433–458, 2015.

G. Ghiani. Arc routing, allocation-arc routing and location-arc routing problems. Technical report, PhD dissertation, Università degli Studi di Napoli Federico II, Italy, 1998.

G. Ghiani and G. Improta. An algorithm for the hierarchical chinese postman problem. *Operations Research Letters*, 26(1):27 – 32, 2000.

G. Ghiani and G. Laporte. Eulerian location problems. *Networks*, 34(4):291–302, 1999.

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

G. Ghiani, G. Improta, and G. Laporte. The capacitated arc routing problem with intermediate facilities. *Networks*, 37(3):134–143, 2001.

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

M. Grötschel and M.W. Padberg. Polyhedral theory. *The traveling salesman problem*, pages 251–305, 1985.

D. Grundel and D. Jeffcoat. Formulation and solution of the target visitation problem. Technical report, Proceedings of the AIAA 1st Intelligent Systems Technical Conference, 2004.

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

S.H. Hashemi Doulabi and A. Seifi. Lower and upper bounds for location-arc routing problems with vehicle capacity constraints. *European Journal of Operational Research*, 224(1):189–208, 2013.

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

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

A. Hildenbrandt and G. Reinelt. Inter programming models for the target visitation problem. *Informatica*, 39:257–260, 2015.

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.

P. Hungerländer. A semidefinite optimization approach to the target visitation problem. *Optimization Letters*, 9(8):1703–1727, 2015. ISSN 1862-4480.

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.

P. Korteweg and T. Volgenant. On the hierarchical chinese postman problem with linear ordered classes. *European Journal of Operational Research*, 169(1):41 – 52, 2006.

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.

A.N. Letchford and R.W. Eglese. The rural postman problem with deadline classes. *European Journal of Operational Research*, 105(3):390–400, 1998.

A.N. Letchford and J.J. Salazar-González. Stronger multi-commodity flow formulations of the capacitated vehicle routing problem. *European Journal of Operational Research*, 244(3):730–738, 2015.

L. Levy and L.D. Bodin. The arc oriented location routing problem. *INFOR.*, 27(1): 74–94, 1989.

H. Longo, M.P. De Aragaõ, and E. Uchoa. Solving capacitated arc routing problems using a transformation to the CVRP. *Computers & Operations Research*, 33(6):1823–1837, 2006.

R.B. Lopes, F. Plastria, C. Ferreira, and B.S. Santos. Location-arc routing problem: Heuristic approaches and test instances. *Computers & Operations Research*, 43:309–317, 2014.

R. Martí and G. Reinelt. *The linear ordering problem: exact and heuristic methods in combinatorial optimization*, volume 175. Springer Science & Business Media, 2011.

H. Min, V. Jayaraman, and R. Srivastava. Combined location-routing problems: A synthesis and future research directions. *European Journal of Operational Research*, 108(1):1–15, 1998.

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

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.

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.

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

G. Nagy and S. Salhi. Location-routing: Issues, models and methods. *European Journal of Operational Research*, 177(2):649–672, 2007.

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

C.S. Orloff. On general routing problems: Comments. *Networks*, 6(3):281–284, 1976.

W.L. Pearn, A.A. Assad, and B.L. Golden. Transforming arc routing into node routing problems. *Computers & Operations Research*, 14(4):285–288, 1987.

A. Pia and C. Filippi. A variable neighborhood descent algorithm for a real waste collection problem with mobile depots. *International Transactions in Operational Research*, 13(2):125–141, 2006.

C. Prodhon and C. Prins. A survey of recent research on location-routing problems. *European Journal of Operational Research*, 238(1):1–17, 2014.

M.A. Salazar-Aguilar, A. Langevin, and G. Laporte. The synchronized arc and node routing problem: Application to road marking. *Computers & Operations Research*, 40 (7):1708–1715, 2013.

E.J. Willemse and J.W. Joubert. Applying min-max k postmen problems to the routing of security guards. *The Journal of the Operational Research Society*, 63:245–260, 2012.

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

L.A. Wolsey. *Integer Programming*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 1998.

L.A. Wolsey and G.L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2014.

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

List of Abbreviations

ARPs	**A**rc **R**outing **P**roblems
BIP	**B**inary **I**nteger **P**rogram
CARP	**C**apacitated **A**rc **R**outing **P**roblem
CO	**C**ombinatorial **O**ptimization
CPP	**C**hinese **P**ostman **P**roblem
C-TVRPP	**C**lustered- **T**arget- **V**isitation **R**ural **P**ostman **P**roblem
HCPP	**H**ierarchical **C**hinese **P**ostman **P**roblem
IP	**I**nteger **P**roblem
LARPs	**L**ocation **A**rc **R**outing **P**roblems
LOP	**L**inear **O**rdering **P**roblem
LP	**L**inear **P**rogramming
LRPs	**L**ocation **R**outing **P**roblems
MC-MDRPP	**M**in **C**ost **M**ulti- **D**epot **R**ural **P**ostman **P**roblem
MDARPs	**M**ulti- **D**epot **A**rc **R**outing **P**roblems
MDRPP	**M**ulti- **D**epot **R**ural **P**ostman **P**roblem
MILP	**M**ixed **I**nteger **L**inear **P**rogramming
MM-K-RPP	**M**in **M**ax **K**- **R**ural **P**ostman **P**roblem
MM-MDRPP	**M**in **M**ax **M**ulti- **D**epot **R**ural **P**ostman **P**roblem
MST	**M**inimum **S**panning **T**ree
RPP	**R**ural **P**ostman **P**roblem
TVARPs	**T**arget- **V**isitation **A**rc **R**outing **P**roblems
TVPs	**T**arget- **V**isitation **P**roblems
TVRPP	**T**arget- **V**isitation **R**ural **P**ostman **P**roblem