Dpto. de Matemática Aplicada II, Universidad de Sevilla
Instituto de Matemáticas de la Universidad de Sevilla (IMUS)
Programa de doctorado "Matemáticas"

Université Libre de Bruxelles
Faculté des Sciences
Doctorat en Sciences

PhD Dissertation

Exact and metaheuristic approaches for Network Design Problems

Author
Natividad González Blanco

Advisors
Prof. Dr. *Bernard Fortz*
Prof. Dr. *Juan A. Mesa López-Colmenar*

November 2022

A mis padres y a mi hermana.

A todas esas personas que me han hecho florecer con su risa.

Resumen

Como consecuencia de la globalización, las interacciones entre países, empresas, personas, etc., se han incrementado durante las últimas décadas. La estructura básica de soporte matemático para modelar interacciones es la red. Por lo tanto, el Diseño de Redes juega un papel importante para facilitar tales interacciones. Sin embargo, la mayoría de los problemas de Diseño de Redes son difíciles de resolver. Nuestro objetivo es estudiarlos desde el punto de vista de la Optimización Combinatoria para encontrar enfoques exactos y metaheurísticos que mejoren el proceso de obtención de soluciones óptimas, o al menos buenas, para aplicaciones prácticas.

En esta tesis estudiamos algunos problemas de Diseño de Redes, los cuáles se pueden agrupar en dos grandes clases detalladas más adelante según la característica principal de cada uno. En todos los problemas se considera que la demanda está dada por un conjunto de pares de puntos origen/destino. Es decir, cada demanda tiene que pasar de un nodo de origen a un nodo de destino. Estos problemas diseñan una red para servir a cierta parte de este conjunto de pares de demanda. Otra característica que comparten es la de la existencia de una red alternativa que puede ser utilizada por este conjunto de demanda. De esta situación se ha destacado la posible competencia existente entre la red que se va a diseñar y la red alternativa actual.

Por un lado, los Capítulos 2, 3 y 4 tratan con problemas de Cobertura para el Diseño de Redes. Estos problemas buscan diseñar una red de tal forma que se maximice la proporción de la demanda cubierta o se cubra un cierto porcentaje del total. En particular, el tercer capítulo muestra una aplicación práctica para el Diseño de Redes en el área del transporte.

Por otro lado, el Capítulo 5 extiende las nociones de λ-Cent-Dian y Centro-Generalizado, ya existentes en Teoría de Localización de Instalaciones, al área de Diseño de Redes. Los problemas tratados en este capítulo están enfocados para diseñar redes que minimicen la distancia máxima de un conjunto de pares de demanda conocido (dentro de esa red), la distancia promedio, una combinación lineal de ambos objetivos o la diferencia entre ellos. Estos objetivos pueden ser de interés para algunas de las necesidades existentes en la actualidad.

Todos los problemas han sido abordados desde el punto de vista de la Programación Matemática. Cada uno de ellos ha sido descrito en detalle y se han encontrado algunas propiedades. Luego, se han propuesto formulaciones. Posteriormente, desde una perspectiva computacional, se han desarrollado métodos de preprocesamiento antes de centrarnos en los procedimientos de resolución.

La investigación aquí realizada también se puede agrupar por la naturaleza de los métodos de resolución utilizados para abordar los problemas propuestos. Por un lado, se han desarrollado algunas estabilizaciones para el método de descomposición de Benders. Por otro lado, también se han considerado enfoques metaheurísticos para algunos de los problemas en cuestión. Se han evaluado procedimientos elaborados por otros autores: un GRASP (Greedy Randomized Adaptive Search Procedure), de tipo búsqueda adaptativa con aleatoriedad, y un algoritmo genético. Además, hemos desarrollado un algoritmo de recocido simulado (Simulated Annealing) y otro algoritmo de búsqueda adaptativa en el caso en el que se consideran vecindarios de gran tamaño (Adaptive Large Neighborhood Search).

Abstract

As a consequence of globalization, interactions among countries, companies, people, etc, have been increased during recent decades. The basic mathematical support structure for modeling such interactions is a network. Thus, Network Design plays an important role for facilitating the interactions. Nevertheless, the majority of Network Design problems are difficult to solve. Our goal is to study them from the Combinatorial Optimization point of view to find exact and metaheuristic approaches that improve the process of getting optimal ,or at least good, solutions for practical applications.

In this thesis, we study some Network Design problems, which can be grouped into two large classes detailed below, according to the main feature of each one. In all the problems it is considered that the demand is given by a set of pairs of origin-destination points. That is, each demand has to move from an origin-node to a destination-node. Another feature in common is the existence of an alternative network that can be used by the demand set. In this situation, the possible existing competition between the network to be designed and the already actual alternative network has been highlighted.

On the one hand, Chapters 2, 3 and 4 deal with Covering Network Design problems. These problems seek to design a network in such a way that the proportion of demand covered is maximized or exceeds a certain percentage of the total. Particularly, the third chapter shows a practical application for Network Design in the transportation area.

On the other hand, Chapter 5 extends the existing notions in Facility Location Theory of λ-Cent-Dian and Generalized-Center to the area of Network Design. The problems in this chapter are focused on designing a network that minimizes the maximum distance of a known set of origin-destination pairs (within that network), the average distance, a linear combination of both objectives or the difference between them. These objectives may be of interest for some of the needs at the present time.

All problems are approached from the standpoint of Mathematical Programming. Each of them has been described in detail and some properties have been found. Then, formulations have been proposed. Afterwards, from a computational perspective, preprocessing methods have been developed before focusing on the resolution procedures.

The research done can also be grouped by the nature of the resolution methods used to tackle the problems proposed. On the one hand, some stabilizations for the Benders decomposition method have been developed. On the other hand, metaheuristic approaches have been also considered for some of the problems concerned. In this situation, Greedy Randomized Adaptive Search Procedures and a Genetic Algorithm elaborated by other authors are evaluated. Furthermore, we have developed a Simulated Annealing and an Adaptive Large Neighborhood Search routine.

Contents

Chapter 1

Introduction

At present, and for some decades now, there has been a growing need to set connections in different aspects of current life, both from an individual and a global point of view. The mathematical structure that models the interaction (connection) between pairs of elements of a set is the graph. If the elements of a graph, either the vertices and/or the edges, have attached weights, the graph is usually called a network. Several features of the system depend on the way the vertices are connected. Problems on how to decide which edges select from an underlying network in accordance with some or several criteria are called Network Design problems. These problems lie in the Discrete Mathematics area and usually are very difficult to solve, but very important in many subjects. Therefore, in the area of Mathematical Optimization, the study of Network Planning and Design began to gain importance.

The main objective of this chapter is to briefly introduce some Network Design problems and describe their most outstanding features. There will be a review of problems related mainly to the areas of telecommunications and transportation. To do this, we will start from the beginning, from the existing problems in Facility Location Theory. Such problems sometimes have similar objective functions since their aim is to serve some demand points and, in some way, they could be considered as particular cases of Network Design. Moreover, Facility Location Theory is a subject well developed, and therefore it is a source of inspiration for Network Design problems. Then, we will introduce the notion of Extensive Facility Location since it is a concept interlaced with that of Network Design in which synergies appear.

1.1 Facility Location Problems

Covering problems along with *median* and *center* problems are three classical main branches of Facility Location Theory.

Covering problems in graphs have attracted the attention of researchers since the middle of the last century. As far as the authors are aware the first papers on the Vertex-Covering problem were due to Berge (1957) and Norman and Rabin (1959) in the late 50s. This problem is related to the Set-Covering problem in which a family of sets is given and the minimal subfamily whose union contains all the elements is sought for. In Hakimi (1965) the Vertex-Covering problem was formulated as an integer linear programming model and solved by using Boolean functions. Toregas et al. (1971) applied the Vertex-Covering problem to the location of emergency services. They assumed that a vertex is covered if it is within a given coverage distance. Church and ReVelle (1974) introduced the Maximal-Covering Location problem by fixing the number of facilities to be located. Each vertex has an associated population and the objective is to cover the maximum population within a fixed distance threshold. Since then many variants and extensions of the Vertex-Covering problem and Maximal-

Covering problem have been studied (see García and Marín (2020)).

Center and *Median* problems in graphs and Euclidean spaces were the basis of the beginning of Location Science in the late 50's and 60's of the past century. Whereas median problems aim at maximizing the efficiency of the system, center ones try to maximize the effectiveness or fairness. Given a set of points (demands, customers), either in a graph or in a continuous space with associated weights, the median problems consist in finding one or several weighted points (facilities) so that the normalized sum of the weighted distances of the facilities to the given points would be minimized. However, center problems consist in finding one or several points so that the worst distance from a demand point to its closest facility is minimized. Median notion fits well with problems in which the goal consists in minimizing the cost or maximizing the profit of the system. The goal of the center problems is to minimize the maximum distance. Thus, it fits better to locate emergency facilities where the farthest point must be as close as possible to the facility. However, isolated these objectives do not satisfy the requirements and/or the aim of many problems in which a balance between both of them is desirable.

1.2 Extensive Facility Location and Network Design Problems

Extensive Facility Location

A facility is called *extensive* if it is too large regarding its environment to be represented by one isolated point. An Extensive Facility Location problem in graphs consists of locating a subgraph that optimizes some objective function and satisfies some constraints regarding a known set of demand points, belonging or not to the vertex set of the underlying graph. This demand wants to access the designed subgraph.

Covering problems have been extended to Extensive Facility Location, where a subnetwork is to be selected from an (physical or not) underlying network with the aim of being used by traffic of goods or people. The Extensive Vertex-Covering problem and the Extensive Maximal-Covering problem are extensions of the point Vertex-Covering and Maximal-Covering problems where the number of vertices of the coverage is substituted by the length or other characteristic of the covering subgraph. These problems depend on the type of underlying graph, the location of the demand, the shape of the graph used for covering and its tips (discrete or continuous), the way of covering and other characteristics. Several papers have dealt with Extensive Facility Location with covering objectives (see Mesa (2018) and the references therein).

The λ-cent-dian problems have been extended to Extensive Facility Location.

The first time that a research on λ-cent-dian of extensive facilities was published was in López-de-los Mozos and Mesa (1992), in which some properties for the λ-cent-dian path in a tree network are derived. The paper of Averbakh and Berman (1999) deals with three problems of path location in trees: minimization of a convex combination of the maximum and average distances, and the two of optimizing one criterion subject to a value for the other. Authors solved all the problems presented in $O(n)$ time by the application of some dynamic programming ideas. Problems which locates a path or a tree-graph with a possibly limited length so that the maximum distance or the sum of distances to the vertices is minimized generalize, to non-point facilities, the concept of center and median, respectively. In these papers Extensive Facility Location problems with other objective functions as *k-centrum*, *ordered median* and *equity measures* have also been reviewed. See for this Mesa and Boffey (1996) and Puerto et al. (2018).

Network Design

Network Design is a broad and spread subject whose models often depend on the field in which they are applied. A classification of the basic problems of Network Design was done by Magnanti and Wong (1984) where some classical graph problems such as the Minimal Spanning Tree, Steiner Tree, Shortest Path, Facility Location and Traveling Salesman problems are included as particular cases of a general mathematical programming model. Since the construction of a network often costs a large amount of money and time, decisions on Network Design constitute a crucial step when planning networks. Infrastructure Network Design is a major step in the planning of a network since the performance, the efficiency, the robustness and other features strongly depend on the selected nodes and the way of connecting them.

Network Design is applied in a wide range of fields: transportation, telecommunications, energy, supply chain, geostatistics, evacuation, monitoring, etc, but especially for the first two named. In the telecommunications area, a very common problem is to design the backbone and local area networks to serve the demand of a set of given points. In these cases, the demand is not given by isolated points, but by pair of points thus generating flows that must be carried out through the network to be designed. In Forsgren and Prytz (2008) it is considered an overview of the issues that arise as well as a number of specific optimization models and problems in this area. In Smith and Winter (1991) there is a set of 7 issues that basically collects problems in the area of telecommunications and other related combinatorial problems. Chapter 2 of Yaman (2006) gathers some significant examples of location problems that arise in the design of telecommunications networks and reviews the polyhedral properties of these problems. Also, Wong (2021) summarizes the major developments in telecommunications network design technologies/services. In transportation, there are many applications, such as, air transportation, postal delivery systems, service networks, trucking, and transit systems. For instance, the main purpose of a rapid transit network is to improve the mobility of the inhabitants of a city or a metropolitan area from a set of origin points to a set of destination points. In this case, the demand is also given by pair of points. This improvement could lead to lower journey times, less pollution and/or less energy consumption which drive the communities to more sustainable mobility. Recently, it has been published the book Crainic et al. (2021), which covers most of the problems in this area. However, the majority of them dealt with in this book have as objective function the cost of constructing the network. In Chapter 17 (see Mauttone et al. (2021)) problems minimizing the travel time of the passengers are also formulated and studied. In Guihaire and Hao (2008) a global review of the crucial strategic and tactical steps of transit planning is presented. They propose a classification of 69 approaches dealing with the design, frequency setting and timetabling of transit lines and their combinations.

Sometimes telecommunications and transportation problems have been classified as hub location problems (see Contreras (2021), Klincewicz (1998) and Martins de Sá et al. (2015)).

Synergies

Extensive Facility Location and Network Design are interlaced subjects where synergies often appear. Moreover, Network Design and point-location problems are very related since the network is the way to access the points and, therefore the efficiency of the optimal points depends on the network design. In fact, several relatively recent models integrate the location of facilities (points) and the design of the network to use to fulfill the demand. The objective function of the model in the paper by Melkote and Daskin (2001) includes transportation cost, facility location and construction of the network.

Direct and indirect coverage

As it is explained in Hutson and ReVelle (1989), the concept of coverage has been used to indicate proximity to a facility within a specified time or distance standard. For network problems, there are two types of coverage that are relevant. Under *direct coverage*, a node is said to be covered only if an arc chosen to be in the network is incident to that node. That is, nodes that are part of the network are covered directly. *Indirect coverage* is more closely aligned with the traditional definition of coverage based on proximity to a facility and includes direct coverage as a special case when the current distance equals zero.

Problems with indirect coverage are closely related to those of Extensive Facility

Location since in the latter there is a set of demand points that is not contained in the set of nodes of the network designed. That is, the demand wants to access the designed subgraph. Furthermore, the notion of indirect coverage is also associated with hubs problems. Hubs, or central trans-shipment facilities, allow the construction of a network where large numbers of direct conections can be replaced with fewer indirect connections (see O'Kelly and Miller (1994)).

Pairs of points of demand

Sometimes, when designing an infrastructure network, the demand is given by pairs of origin/destination points, called O/D pairs, and each such pair has an associated weight representing the traffic between the origin and the destination nodes. Usually, this demand is encoded using an origin/destination matrix. The Network Design problems presented in this dissertation support this type of demand.

Alternative networks

When planning a new network, often there exists at least a network already functioning. If such a network is offering its service to the same set of origin/destination pairs as those considered for the new network, in some way both modes compete with each other. For example, a new rapid transit system may be planned in order to improve the mobility of a big city or metropolitan area, in which there already exists another transit system, in addition to the private transportation system. This current transit system could be denser than the planned one but slower since it uses the same right-of-way as the private traffic system. Thus, in some way, both systems compete with each other and both compete with the private system of transportation. A similar effect occurs with mobile telecommunication operators. Therefore, the traffic between an origin and a destination is distributed among the several companies or systems that provide the service.

Researches in Facility Location and Extensive Facility Location consider a customer-server system, where demand uses the network to access the point or extensive facilities. However, in some Network Design problems to reach or be close to the facility is not enough to complete the service. In many telecommunication, transportation, public services and other systems the demand uses the facility as an intermediate instead of a final step (see Contreras and Fernández (2012)). These problems consider a customer-server-customer system. This is, for example, the case of the public urban transportation network, where a customer has to spend a time to reach the stop/station from its origin, then wait for the next service, spend an in-vehicle time, and finally reach its destination. The network design of these systems is even more complex than those where the customers just have to go to the servers. In this situation, the customer has used at least two types of systems and they are not competing between them. That is, for some customers, they are, in fact, cooperating.

In this thesis, we have studied the situation in which there is competition between the network that is going to be designed and another that already exists.

Mode sharing

Given a set of O/D pairs, there are mainly two ways of allocating the share of each system. The first one is the binary all-or-nothing way where each demand is only assigned to one of the proposed modes. In this way, for the case in which there are two modes, there will be a set of O/D pairs assigned to the new network and the remaining set assigned to the existing network, such as in Perea et al. (2020). The second one is based on some continuous function, using, for example, a multi-logit probability distribution, as in Cascetta (2009). In this case, each demand is shared between the different systems. Both allocation schemes are based on the comparison of distances, times, costs, generalized costs or utilities. In this thesis, we consider the binary one, where each O/D pair is assigned to one of the available modes.

Optimality measures considered for Network Design

In Infrastructure Network Design, since it is generally too expensive to connect all the potential nodes, one must determine a subnetwork that serves at best the traffic demand. Depending on the application, different optimality measures can be considered. In particular, in the field of transportation, and especially in the area of passenger transportation, the aim is to get the infrastructure close to potential customers. In this framework, Schmidt and Schöbel (2014) propose to minimize the maximum routing cost for a set of origin/destination pairs when using the new network to be constructed. Alternatively, the traffic between an origin and a destination may be considered as captured if the cost or travel time when using the network is not larger than the cost or travel time of the best alternative solution (not using the new network). In this case, Perea et al. (2020) and García-Archilla et al. (2013) propose to select a subnetwork from an underlying network with the aim of capturing or covering as much traffic as possible for a reasonable construction cost. On the other hand, two of the most present objectives in the literature when planning a telecommunications network are satisfying all projected demand at minimum total discounted cost (investment plus operating), or selectively satisfying demand to maximize the total profit (see Balakrishnan et al. (1991) and Ji et al. (2015)).

Computational complexity

Problems with indirect and direct coverage which locate a subtree of a known network with a limited number of vertices or bounded length are NP-hard in general since they are special cases of the Steiner Tree problem (see Johnson and Garey (1979)). However, if the underlying graph is a tree some of the problems can be solved by polynomial-time algorithms (see Bern and Bienstock (1991)). In Hakimi et al. (1993), authors determine the algorithmic complexity of four generalization optimization problems that locate paths or tree-shaped facilities in trees and general networks. They are related to the p-center, p-median, p-max-eccentricity and p-max-distance-sum problems for the cases $p = 1$ and $p > 1$. In Puerto et al. (2018), the location of dimensional facilities on networks is reviewed, mainly considering two common objective functions in literature, the center and the median. Moreover, the best-known complexity results are presented. Without loss of generality, in the problems dealt with in this thesis we have considered coverage in its direct form. We have worked with general networks and, as we justify in each chapter, the problems we have studied are NP-hard. Therefore, from a mathematical point of view, it is a simplification. Considering indirect coverage in these problems only adds a bit of complexity to the formulation.

In Bärmann (2016), the author gives a classification of the most important types of Network Design problems. They could be classified into the following four groups with increasing complexity. Problems without capacity on the edges, named as Uncapacitated Network Design problems. Problems with a fixed charge on each arc to determine the price to equip it with a capacity, named as Fixed-Charge Capacitated Network Design problems. The demand of each O/D pair is given since the aim is to provide enough arcs to route the requested demand through the network. In third place, Capacity or Coverage Expansion Network Design problems, which are a generalization of the above. In this situation, there are some previously existing capacities that may be used at no cost. Furthermore, there is a set of available modules for each arc to increase its capacity. Finally, Multiperiod Network Expansion problems, which consider that the demand of each O/D pair is different for each time period in the planning horizon.

At an early stage the NP-hardness of a special Network Design problem was stated. In the paper by Johnson et al. (1978), it was shown that the *Knapsack problem* is reducible to the problem of choosing a subgraph such that it minimizes the sum of the shortest path weights between all vertex pairs subject to a budget constraint. In Perea et al. (2020) it is proved that, considering a budget constraint, the problem to find the optimal network that maximizes the sum of the weights of a set of pairs to cover is NP-hard, regardless of whether cycles are allowed or not.

1.2.1 Description and notation for the elements in Network Design problems

For the network design problems considered in this dissertation, the following elements are considered.

- An undirected graph denoted by $\mathcal{N} = (N, E)$, where N and E are the sets of potential nodes and edges that can be constructed. Then, $\mathcal{N}$ is the *potential graph*. Each element $e \in E$ is denoted by $\{i, j\}$, with $i, j \in N$. We use the notation $i \in e$ if node i is a terminal node of e. Besides, for each $e = \{i, j\} \in E$, we define two arcs: $a = (i, j)$ and $\hat{a} = (j, i)$. The resulting set of arcs is denoted by A. Furthermore, an *alternative* directed *graph* $\mathcal{N}^{alt} = (N^{alt}, A^{alt})$, with $N^{alt} = N$ and $A^{alt} \subset N \times N$ such that $A^{alt} \neq A$, is present. To simplify the notation, in the problem formulations presented in this dissertation we do not refer directly to $\mathcal{N}^{alt}$.

- $W \subset N \times N$ represents a set of origin/destination (O/D) pairs that must be defined. This is a strict inclusion since couples with both equal nodes are not eligible. Each $w = (w^s, w^t) \in W$, with $w^s \neq w^t$, is composed by an origin node $w^s \in N$ and a destination node $w^t \in N$.

- Costs for building nodes, $i \in N$, and edges, $e \in E$, in the potential network $\mathcal{N}$ are denoted by b_i and c_e, respectively, with $b_i, c_e \in \mathbb{R}_{\geq 1}$. We denote the total construction cost as $C_{total} = \sum\limits_{i \in N} b_i + \sum\limits_{e \in E} c_e$.

- The weight of every feasible arc $a \in A$ is denoted by $d_a \in \mathbb{R}_{\geq 1}$. Such parameter may represent length, time or cost of traversing, generalized cost or even utility. Without loss of generality, we will refer to it as the distance between the end-nodes of each arc.

- Each $w = (w^s, w^t) \in W$ has some parameters and elements associated:
  - A demand $g^w \in \mathbb{Z}^+$. We denote the total demand as $G_{total} = \sum_{w \in W} g^w$. Without loss of generality, the demand is concentrated in the set of potential nodes.
  - A length $u^w \in \mathbb{R}^+$, corresponding to an unknown predefined path from w^s to w^t in $\mathcal{N}^{alt}$.
  - For a given subnetwork $\mathcal{S} \subseteq \mathcal{N}$, $d_{\mathcal{S}}(w)$ expresses the length of the shortest path from w^s to w^t in the subnetwork $\mathcal{S}$. If pair w cannot be connected within $\mathcal{S}$, we define $d_{\mathcal{S}}(w) = +\infty$. To name the notion of the minimum value between $d_{\mathcal{S}}(w)$ and u^w, we use $\ell_{\mathcal{S}}(w)$.

  - A subgraph $\mathcal{N}^w = (N^w, E^w) \subseteq \mathcal{N}$ containing all *feasible nodes* and *edges* for w, i.e. that belong to a path in $\mathcal{N}$ whose total length is lower than or equal to u^w. We also denote A^w as the set of *feasible arcs* in $\mathcal{N}^w$. In Section 2.2.2, we describe how to construct these subgraphs. The meaning of feasible node, edge and/or arc can be further restricted depending on the conditions of the problem to be studied.

- For the neighborhood purpose, let $\delta(i)$ be the set of edges incident to node i. By $\delta^w(i)$ is denoted the set of edges incident to node i in the subgraph $\mathcal{N}^w$. Notation $\delta^w_+(i)$ ($\delta^w_-(i)$ respectively) is used to refer to the set of arcs going out (in, respectively) of node $i \in N^w$. In particular, we set $\delta^w_-(w^s) = \emptyset$ and $\delta^w_+(w^t) = \emptyset$.

The existence of $\mathcal{N}^{alt}$ translates into the fact that there already exists a different network competing with the one to be constructed, named as the *prospective network*. It is considered an all-or-nothing way. That is, each demand goes through one of them, it cannot be decomposed. In that sense, each length u^w performs as a threshold to get paths shorter than it in $\mathcal{N}$. We refer to them as *competitive paths*. If the O/D pair w is assigned to the prospective network, we then say that it is *covered* or *served*. For example, in terms of the transportation area, the existing network $\mathcal{N}^{alt}$ represents a private transportation mode, the planned one $\mathcal{N}$ represents a public transportation mode and the parameters g^w, u^w, $w \in W$, refer to the passengers going from w^s to w^t and the utility of taking the private mode. The O/D pair w is supposed to travel using the private mode if there is not a path in the public network with better utility than u^w.

1.3 Decomposition Methods

To solve large scale mixed-integer linear optimization problems it is often useful to rely on decompostion methods. Lagrangean relaxation, Benders decomposition and column generation constitute the most popular approaches of this type.

Lagrangian methods are relaxation methods that approximate a difficult problem to solve with a simpler one, using Lagrange multipliers associated to some constraints. The solution to the relaxed problem is an approximation of the original problem and it provides useful information. Lagrangian techniques are still very important today (see Guignard (2003) and Fathollahi-Fard et al. (2020)).

By using the Benders decomposition approach, the formulation is decomposed into two sets of variables, such as one of them is formed only by continuous variables. The rest of them contains a set of *complicating variables*. Usually, complicating means integer variables. The idea of this procedure consists in isolating such set of variables. That is, this procedure relies on the projection of the feasible set of the problem onto the space that contains the complicating variables. By projecting out variables, the total number of variables is decreased, significantly in most cases. This decrease translates into the addition of new linear constraints which could be necessary to describe the projection. In this procedure, continuous variables are projected out and new linear constraints, formed by the complicating variables, are added. Hence, this decomposition method is more appropriate for problems that have a greater number of variables than constraints. In this thesis, we have chosen the Benders decomposition theory as the tool to solve the problems that concern us. Their formulations can be decomposed in the same way as the Benders decomposition method requires. We can check that the formulations presented in this dissertation are composed of a similar number of variables and constraints. Nevertheless, this has not turned out to be an inconvenience, since not the entire set of constraints generated by the Benders procedure is used. In Subsection 2.1, the Benders decomposition framework is briefly described.

Column generation is another often-applied approach. Normally, it is used in combination with a Dantzig-Wolfe reformulation of the original problem. Dantzig-Wolfe decomposition is the dual of Benders decomposition. Then, the formulation is decomposed into two sets of constraints.

The last two approaches have been gaining interest over the years as an alternate automatic scheme but remains infamously difficult to use efficiently since they are hard to implement, but also due to the fact that to use them the formulation of the problems needs to exhibit a specific structure. For more details about these three categories of exact methods see Martin (2012).

Although decomposition methods cannot solve nowadays large instances in a short time, this is not necessarily a disadvantage, it depends on the problem to be solved. For example, in transportation area, the Infrastructure Network Design problems belong to the strategic phase of the sequential process of transportation planning. Therefore, the computational time is less important in this phase than in others, such as operational or online phases. Moreover, decomposition methods require profound knowledge and thus provide more insight on the problems, the mathematical formulations and the solutions obtained.

1.4 Metaheuristic solution algorithms

Since most Network Design problems are NP-hard, some research efforts have been oriented to apply metaheuristic algorithms to obtain good though not necessarily optimal solutions, within a reasonable computational time. For instance, in the fields of transportation and telecommunications Network Design, Tabu Search (Pedersen et al. (2009), Xu et al. (1996)), Genetic Algorithms (Król and Król (2019), Perea et al. (2020), Chou et al. (2001)), Simulated Annealing (Fan and Machemehl (2006), Kermansshahi et al. (2010), Girgis et al. (2014)), Greedy Randomized Adaptive Search Procedures (García-Archilla et al. (2013), Maya Duque and Sörensen (2011), Risso and Robledo (2013)) and Adaptive Large Neighborhood Search algorithms (Canca et al. (2016), Canca et al. (2017), Zhang et al. (2022), Mehta et al. (2015)) have been applied to medium-size instances. Besides, matheuristics procedures have been also exploited (see Canca et al. (2019), Chouman and Crainic (2015), Wu et al. (2020)).

1.5 Contributions of this thesis

Chapter 2 presents several contributions. It is based on the paper by Víctor Bucarey, Bernard Fortz, Natividad González-Blanco, Martine Labbé and Juan Antonio Mesa López-Colmenar (see Bucarey et al. (2022)). First, we present new mathematical integer formulations for the Covering Network Design problems (MC) and (PC). The demand is given by a set of origin/destination pairs, instead of single demand points. The formulation for (MC) is stronger than a previously proposed one, see e.g. Marín and Jaramillo (2009) and García-Archilla et al. (2013) (although the proposed formulation was not the main purpose of the latter), while (PC) was never studied to the best of our knowledge. Our second contribution consists of polyhedral properties that are useful from the algorithmic point of view. A third contribution is the study of decomposition algorithms for the Network Design based on different Benders implementations. We propose a normalization technique and we consider the facet-defining cuts. Our computational experiments show that our Benders implementations are competitive with exact and non-exact methods existing in the literature and even comparing with the Benders decomposition existing in `CPLEX`.

In Chapter 3, we have applied our most competitive implementation from Chapter 2 to a formulation based on the integrated design of a rapid transit line and the relocation of a slow transit line. This work is described in the short-paper by Natividad González-Blanco, Antonio J. Lozano, Vladimir Marianov and Juan A. Mesa (see González-Blanco et al. (2021)).

In Chapter 4, we review some literature and verify that many metaheuristics have been developed to solve very specific problems in the area of transportation and telecommunications Network Design. The (MC) problem of Chapter 2 is relatively well-known in the Network Design area. However, to the best of our knowledge, it seems that it has been somewhat forgotten when it comes to applying the classical metaheuristics that exist in the literature. This problem has only been studied from two metaheuristic points of view, one in García-Archilla et al. (2013) and another in

Perea et al. (2020). Then, two different techniques from those already discussed in these mentioned works have been proposed in this chapter. We have developed two Neighborhood Search procedures. One of them is a Simulated Annealing algorithm and the other is an Adaptive Large Neighborhood Search procedure. A difference between them is that the second one works with a larger neighborhood. Nevertheless, as we will see, not always working with a large neighborhood will achieve better results. An exhaustive analysis has been elaborated for each one, considering also the two metaheuristics already studied mentioned above. These metaheuristics are designed to obtain good though not necessarily optimal solutions, within a reasonable computing time.

Finally, Chapter 5 extends for the first time the existing solution concepts in Facility Location Theory of λ-cent-dian and generalized-center to the much more complex Network Design area. Such as in the previous chapters, we are given the demand as a set of origin/destination pairs, instead of single points, and a budget constraint. The λ-cent-dian concept aims at studying the trade-off between efficiency and equity. The generalized-center is the particular case of the λ-cent-dian concept when $\lambda \to \infty$ and it consists of the difference between these both measures of efficiency and equity. We investigate the properties of the λ-cent-dian and generalized-center solution networks under the lens of equity, efficiency, and Pareto-optimality with respect to the bicriteria center/median problem, obtaining some similar conclusions than in the case of locating a facility. As it happens in Facility Location, we prove that in Network Design area the Pareto-optimality solutions set is not always completely generated with minimization of the λ-cent-dian function with $\lambda \in [0, 1]$. Secondly, we provide a mathematical formulation for the λ-cent-dian problem with $\lambda \geq 0$. Besides, we discuss the bilevel structure of this problem for $\lambda > 1$ and we readapt the formulation for this case. A third contribution is to describe a procedure to give a complete parametrization of the Pareto-optimality set based on solving two linear formulations. Then, we evaluate and discuss the quality of the different solution concepts considered using some inequality measures, not only from the point of view of efficiency and equity. In this evaluation we include the generalized-center notion, which has not been taken into account in many works. Eventually, for $\lambda \in [0, 1]$ we test both a Benders decomposition method and a metaheuristic technique to solve it at scale.

Chapter 2

Benders Decomposition for Network Design Covering Problems

We consider two variants for the Covering Network Design problem. We are given a set of origin/destination pairs, called O/D pairs, and each such O/D pair is covered if there exists a path in the network from the origin to the destination whose length is not larger than a given threshold.

After presenting formulations, we develop a Benders decomposition approach to solve the problems. Further, we consider several stabilization methods to determine Benders cuts as well as the addition of cut-set inequalities to the master problem. We also consider the impact of adding an initial solution to our methods. Computational experiments show the efficiency of these different aspects.

2.1 Introduction

Covering problems in graphs and extensions

Covering problems in graphs have attracted the attention of researchers since the middle of the last century. As far as the authors are aware the first papers on the Vertex-Covering Problem were due to Berge (1957) and Norman and Rabin (1959) in the late 50s. This problem is related to the Set-Covering Problem in which a family of sets is given and the minimal subfamily whose union contains all the elements is sought for. In Hakimi (1965) the Vertex-Covering Problem was formulated as an integer linear programming model and solved by using Boolean functions. Toregas et al. (1971) applied the Vertex-Covering Problem to the location of emergency services. They assumed that a vertex is covered if it is within a given coverage distance. Church and ReVelle (1974) introduced the Maximal-Covering Location problem by fixing the number of facilities to be located. Each vertex has an associated population and the objective is to cover the maximum population within a fixed distance threshold. Since then many variants and extensions of the Vertex-Covering and Maximal-Covering problems have been studied (see García and Marín (2020)). A classical binary formulation of Vertex-Covering problem is the following, denoted as *(VC)*, where b_i, $i \in N$ refers to a vector cost.

$$(VC) \quad \min_{\boldsymbol{y}} \quad \sum_{i\in N} b_i \, y_i \tag{2.1}$$

$$\text{s.t.} \quad y_i + y_j \geq 1, \qquad \{i,j\} \in E, \tag{2.2}$$

$$y_i \in \{0,1\}, \qquad i \in N. \tag{2.3}$$

Covering problems have been extended to Extensive Facility Location, where facilities are too large to be represented as isolated points, and Network Design, where a subnetwork is to be selected from an (physical or not) underlying network with the aim of being used by traffic of goods or people. The Extensive Vertex-Covering and

Extensive Maximal-Covering problems are extensions of the point Vertex-Covering and Maximal-Covering problems where the number of vertices of the coverage is substituted by the length or other characteristic of the covering subgraph. These problems depend on the type of underlying graph, the location of the demand, the shape of the graph used for covering and its tips (discrete or continuous), the way of covering and other characteristics. Several papers have dealt with Extensive Facility Location with covering objectives (see Mesa (2018) and the references therein).

In this chapter we deal with Covering Network Design problems.

For the Covering Network Design problems presented in this chapter the demand is given by pairs of origin/destination points, called O/D pairs. Each such pair has an associated weight representing the traffic between the origin and the destination. Usually, this demand is encoded using an origin/destination matrix. Besides, the existence of an alternative transportation mode is considered. This mode is functioning and offering its service to the same set of O/D pairs. Then, we consider that both modes are competing between them.

Since it is generally too expensive to connect all the potential nodes, one must determine a subnetwork that serves best the traffic demand. Depending on the application, different optimality measures can be considered. In particular, in the field of transportation, and especially in the area of passenger transportation, the aim is to get the infrastructure close to potential customers, but not to *cover* everywhere. In this framework, Schmidt and Schöbel (2014) propose to minimize the maximum routing cost for an origin/destination pair set when using the new network. Alternatively, the traffic between an origin and a destination may be considered as captured if the cost or travel time when using the network is not larger than the cost or travel time of the best alternative solution (not using the new network). In this case, Perea et al. (2020) and García-Archilla et al. (2013) propose to select a subnetwork from an underlying network with the aim of capturing or covering as much traffic as possible for a reasonable construction cost. This chapter is devoted to this problem, called the **M**aximal **C**overing Network Design problem (MC) as well as to the closely related problem called, **P**artial **C**overing Network Design problem (PC). The latter aims to minimize the network design cost for constructing the network under the constraint that a minimum percentage of the total traffic demand is covered.

Covering Network Design problems presented in this chapter only consider the binary all-or-nothing way of assigning the demand. That is, the demand is only covered by one of the proposed modes. Typically, the demand is covered if the demand points are served within a range of quality service, as in Perea et al. (2020). We consider that each O/D pair is covered only if the time spent to travel from its origin to its destination in the network is below a threshold. This threshold represents the comparison between the time spent in the proposed network and a private mode, assigning the full share to the most beneficial one.

Benders Decomposition framework

Benders decomposition was introduced in the early 60s as a method to solve mixed-integer problems (see Benders (1962)).

Benders decomposition for Network Design problems has been studied since the 80s. In Magnanti et al. (1986), the authors minimize the total construction cost of an uncapacitated network subject to the constraint that all O/D pairs must be covered. Given the structure of the problem, the Benders reformulation is stated with one subproblem for each O/D pair. A Benders decomposition for a multi-layer Network Design problem is presented in Fortz and Poss (2009). In Costa et al. (2009), a Multi-Commodity Capacitated Network Design problem is studied and the strength of different Benders cuts is analysed. In Marín and Jaramillo (2009) a multi-objective approach is solved through Benders decomposition. The coverage is maximized and the total cost design is minimized. Benders decomposition was also applied in Botton et al. (2013), in the context of designing survivable networks. In Mahéo et al. (2020), authors present how to apply the Benders scheme to specific problems, with an emphasis on the case in which the subproblem formulation can be decomposed naturally into independent blocks of constraints. After, they explain and justify how to extend it to the case with integer subproblems.

This type of decomposition has been applied to many problems in different fields, see Rahmaniani et al. (2017) for a recent literature review on the use of Benders decomposition in Combinatorial Optimization. One recent contribution applied to Set-Covering and Maximal-Covering Location problems appears in Cordeau et al. (2019). The authors propose different types of normalized Benders cuts for these two covering problems.

This decomposition procedure is briefly explained below, which is also known as a projection algorithm. It consists of a sequence of projections, relaxations and outer approximations.

Definition 1. *Let Q be a set of points $(\boldsymbol{x}, \boldsymbol{z}) \in \mathbb{R}^n \times \mathbb{R}^m$. Then, the projection of Q onto the x-space, denoted by $Proj_x(Q)$, is the set of points given by*

$$Proj_{\boldsymbol{x}}(Q) = \{\boldsymbol{x} \in \mathbb{R}^m : (\boldsymbol{x}, \boldsymbol{z}) \in Q \text{ for some } \boldsymbol{z} \in \mathbb{R}^n\}.$$

The classic structure of a Network Design problem suitable for applying the Benders decomposition method looks like this:

$$\begin{aligned} \min \quad & c^T x + f^T y \\ \text{s.t.} \quad & Ax + By = b \\ & Dy = d \\ & x \in (\mathbb{R}_+ \cup \{0\})^{n_1} \\ & y \in (\mathbb{Z}_+ \cup \{0\})^{n_2} \end{aligned} \tag{2.4}$$

where it is considered that the decision variables have been organized into two sets, $x \in (\mathbb{R}_+ \cup \{0\})^{n_1}$ and complicating variables $y \in (\mathbb{Z}_+ \cup \{0\})^{n_2}$. The latter have to satisfy the constraint set $Dy = d$, with $D \in \mathbb{R}^{m_2 \times n_2}$ and $d \in \mathbb{R}^{m_2}$ given. Besides, both types of variables must satisfy the constraint set $Ax + By = b$, where $A \in \mathbb{R}^{m_1 \times n_1}$, $B \in \mathbb{R}^{m_1 \times n_2}$ and $b \in \mathbb{R}^{m_1}$ are also known. The objective function contains the vectors $c \in \mathbb{R}^{n_1}$ and $f \in \mathbb{R}^{n_2}$.

Taking into account this previous concept, model (2.4) can be re-expressed as

$$\min_{\overline{y} \in Y} \left\{ f^T \overline{y} + \min_{x \geq 0} \{ c^T x \mid Ax = b - B\overline{y} \} \right\} \tag{2.5}$$

where $Y = \{y \mid Dy = d, y \in (\mathbb{Z}_+ \cup \{0\})^{n_2}\}$. That is, the original model is decomposed into two problems. First, we consider the following *master problem*

$$\begin{aligned} \min \quad & f^T y + \mathbf{q}(\overline{y}) \\ \text{s.t.} \quad & D\overline{y} = d \\ & \overline{y} \in (\mathbb{Z}_+ \cup \{0\})^{n_2} \end{aligned} \tag{2.6}$$

where $\mathbf{q}(\overline{y})$ represents the expected value of the objective function of the so-called *Subproblem* or *Slave Problem* given by

$$\begin{aligned} \mathbf{q}(\overline{y}) = \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b - B\overline{y}, \\ & x \in (\mathbb{R}_+ \cup \{0\})^{n_2}. \end{aligned} \tag{2.7}$$

Usually, $\mathbf{q}(\overline{y})$ is referred as the *incumbent.* Note that in formulation (2.7), $\overline{y}$ denotes a feasible solution vector from (2.6). Subproblem (2.7) is a continuous linear problem whose dual problem is

$$\begin{aligned} \max \quad & (b - B\overline{y})^T u \\ \text{s.t.} \quad & A^T u \leq c, \\ & u \in \mathbb{R}^{m_1}. \end{aligned} \tag{2.8}$$

This dual formulation is known as the *Benders Subproblem.* Hence, by (2.8), formulation (2.5) is equivalent to

$$\min_{\overline{y}\in Y}\left\{f^T\overline{y}+\max_{u\in\mathbb{R}^{m_1}}\left\{(b-B\overline{y})^T u \mid A^T u\leq c\right\}\right\}. \tag{2.9}$$

We observe that the feasible space of the *Benders Subproblem*, $\mathcal{S}=\{u \mid A^T u\leq c\}$, is independent from $\overline{y}$. If the space $\mathcal{S}$ is empty, $\mathcal{S}=\emptyset$, which means that this dual problem is infeasible, two situations can be handled: (1) its *Slave Problem* is unbounded and, consequently, the original problem is also unbounded, or (2) its *Slave Problem* is infeasible and, then, the original problem is infeasible. In the case in which $\mathcal{S}\neq\emptyset$ there are two different options too. In the unbounded case, let $\mathcal{R}$ be the set of extreme rays of $\mathcal{S}$ and $\mathcal{U}^{\mathcal{R}}$ be the set of unboundedness directions such that $(b-B\overline{y})^T u^r>0$, $u^r\in\mathcal{U}^{\mathcal{R}}$. As this indicates the infeasibility of the $\overline{y}$ solution for the *Slave Problem*, it has to be avoided. Thus, constraints $(b-B\overline{y})^T u^r\leq 0$, $r\in\mathcal{R}$, are added. In the bounded case, let $\mathcal{P}$ be the set of extreme points of $\mathcal{S}$ and u^p be one of them which is the optimal solution of the inner problem. Therefore, (2.9) can be reformulated as

$$\begin{aligned}\min_{\overline{y}\in Y}\quad & f^T\overline{y}+\max_{p\in\mathcal{P}}\{(b-B\overline{y})^T u^p\}\\ \text{s.t.}\quad & (b-B\overline{y})^T u^r\leq 0,\quad r\in\mathcal{R}\end{aligned} \tag{2.10}$$

By the *Weak Duality Theorem*, we know that $\mathbf{q}(\overline{y})\geq(b-B\overline{y})^T u^p$, $p\in\mathcal{P}$. Then, without fixing the vector of variables y, (2.10) is equivalent to

$$\begin{aligned}\min_{y,\mathbf{q}(y)}\quad & f^T y+\mathbf{q}(y)\\ \text{s.t.}\quad & \mathbf{q}(y)\geq(b-By)^T u^p,\quad p\in\mathcal{P}\\ & 0\geq(b-By)^T u^r,\quad r\in\mathcal{R}\\ & Dy=d\\ & y\in(\mathbb{Z}\cup\{0\})^{n_2},\quad \mathbf{q}(y)\in\mathbb{R},\end{aligned} \tag{2.11}$$

which is known as *Benders master problem.* The first two blocks of constraints are referred to as *Optimality* and *Feasibility cuts.* Both types of cuts are known as *Benders cuts.* These blocks are considerably large and it is not practical to use them completely. Thus, the idea is that the algorithm solves a *"relaxed" Benders master problem* which includes a subset of constraints of these large blocks. For that, the algorithm starts by solving the master problem which either does not include initial Benders cuts or includes few. Then, the solution obtained is used to fix the corresponding parameters in the Benders Subproblem and solve it. The dual solution obtained represents a potential improvement for the variables of the master problem and is used in the form of a optimality or feasibility cut. Then, the augmented master problem is solved again. To reach optimality, the algorithm continues until the objective value of the Subproblem and the incumbent are equal.

Nowadays, there are two ways of implementing a Benders decomposition scheme. On the one hand, using a cutting plane scheme, which is the classical one, but seldom used in practice. In each iteration a Benders cut is generated to be added to the master problem and solve it again as an ILP. On the other hand, as other authors have already noted, the most efficient way in which this strategy is used is together with the branch-and-cut scheme. That is, at each node of the branching tree of the branch-and-cut scheme Benders cuts (among other) can easily be generated and embedded into it. This prodedure is named as the *branch-and-Benders-cut* scheme.

Structure of this chapter

The structure of the chapter is as follows. In Section 2.2, we present mixed-integer linear formulations for (MC) and (PC). We also study some polyhedral properties of the formulations and propose a simple algorithm to find an initial feasible solution for both problems. In Section 2.3, we study different Benders implementations and some algorithmic enhancements. Also, we discuss some improvements based on cut-set inequalities. A computational study is detailed in Section 2.4. Finally, our conclusions are presented in Section 2.5.

2.2 Problem formulations and some properties

In this section we present mixed-integer linear formulations for the *Maximal-Covering Network Design* Problem (MC) and the *Partial-Covering Network Design* Problem (PC). We also describe some preprocessing procedures. We finish with some polyhedral properties.

According to the elements defined in Subsection 1.2.1, for both Mixed-Integer Linear Formulations, the following binary variables are used:

- Node selection variables. For each $i \in N$, y_i is a binary variable to decide whether or not node i is built.
- Edges selection variables. For each $e \in E$, x_e is a binary variable to decide whether or not edge e is built.
- Mode choice variables. For each $w \in W$, z^w is a binary variable that takes value 1 if the O/D pair w is covered.
- Flow variables. For each $w \in W$, they are used to model a path from w^s to w^t in the network to be built, if possible. Variable $f_a^w, a \in A$ takes value 1 if arc a belongs to such path for w.

2.2.1 Mixed-Integer Linear Formulations

We first present a formulation of the Maximal-Covering Network Design Problem (MC), whose aim is to design an infrastructure network maximizing the total demand covered subject to a budget constraint:

$$(MC) \max_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{f}} \sum_{w\in W} g^w z^w \tag{2.12}$$

$$\text{s.t.} \sum_{e\in E} c_e x_e + \sum_{i\in N} b_i y_i \leq \alpha\, C_{total}, \tag{2.13}$$

$$x_e \leq y_i, \qquad e\in E, i\in e, \tag{2.14}$$

$$\sum_{a\in\delta_+^w(i)} f_a^w - \sum_{a\in\delta_-^w(i)} f_a^w = \begin{cases} z^w, & \text{if } i = w^s, \\ -z^w, & \text{if } i = w^t, \\ 0, & \text{otherwise,} \end{cases} \quad w\in W, i\in N^w, \tag{2.15}$$

$$f_a^w + f_{\hat{a}}^w \leq x_e,\ w\in W,\ e=\{i,j\}\in E^w : a=(i,j),\ \hat{a}=(j,i), \tag{2.16}$$

$$\sum_{a\in A^w} d_a f_a^w \leq u^w z^w, \qquad w\in W, \tag{2.17}$$

$$y_i,\ x_e,\ z^w \in \{0,1\}, \qquad i\in N,\ e\in E^w,\ w\in W, \tag{2.18}$$

$$f_a^w \in \{0,1\}, \qquad a\in A^w, w\in W. \tag{2.19}$$

The objective function (2.12) to be maximized represents the demand covered. Constraint (2.13) limits the total construction cost, being $\alpha \in (0,1]$. If we consider $\alpha = 0$ the optimal solution network is to construct nothing, no pair is covered. Constraint (2.14) ensures that if an edge is constructed, then its terminal nodes are constructed as well. For each pair w, expressions (2.15), (2.16) and (2.17) guarantee demand conservation and link flow variables f_a^w with mode choice variables z^w and design variables x_e. Constraints (2.16) are named *capacity constraints* and they force each edge to be used only in one direction at most. Constraints (2.17), referenced as *mode choice constraints*, put an upper bound on the length of the path for each pair $w = (w^s, w^t)$. This ensures variable z^w takes value 1 only if there exists a path between w^s and w^t with length at most u^w in the prospective network. This path is represented by variables f_a^w. Finally, constraints (2.18) and (2.19) state that variables are binary.

In Perea et al. (2020), authors prove that this problem is NP-hard.

Remark 1. *For (MC), note that if $\alpha = 1$, there is not a limit of budget. The problem consists of a search for feasible paths.*

The other formulation presented is for the Partial-Covering Network Design problem (PC), which minimizes the total construction cost of the network subject to a minimum coverage level of the total demand and can be formulated as follows:

$$
\begin{aligned}
(PC) \quad \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{f}} \quad & \sum_{i \in N} b_i y_i + \sum_{e \in E} c_e x_e && (2.20)\\
\text{s.t.} \quad & \sum_{w \in W} g^w z^w \geq \beta \, G_{total}, && (2.21)\\
& \text{Constraints (2.14), (2.15), (2.16), (2.17), (2.18), (2.19),}
\end{aligned}
$$

where $\beta \in (0, 1]$. If we consider $\beta = 0$ the optimal solution network is to construct nothing. The objective function (2.20) to be minimized represents the design cost. Constraint (2.21) imposes that, at least, a proportion β of the total demand is covered. It is named as *trip coverage constraint.*

Proposition 1. *For the specific case $\beta = 1$, (PC) is an extension of the Vertex-Cover problem* (2.1)-(2.3).

Proof. The (PC) can be rewritten in the following manner. For each $w \in W$, let L^w be a set with all its competitive paths (that is, shorter than u^w) in $\mathcal{N}$. Besides, each competitive/feasible path $l \in L^w$ has an associated cost $a_l^w \in \mathbb{R}_{\geq 1}$ and a binary variable h_l^w, which takes value 1 if the feasible path l is assigned to w.

$$
\begin{aligned}
\min_{\boldsymbol{h}} \quad & \sum_{w \in W} \sum_{l \in L^w} a_l^w \, h_l^w && && (2.22)\\
\text{s.t.} \quad & \sum_{l \in L^w} h_l^w \geq 1, && w \in W, && (2.23)\\
& h_l^w \in \{0, 1\}, && l \in L^w, w \in W. && (2.24)
\end{aligned}
$$

For each feasible path $l \in L^w$, binary variable h_l^w is defined, which takes value 1 if l is the feasible path assigned to w. □

The following remarks are related to the flow variables set of both previous formulations.

Remark 2. *For (MC) and (PC), given a design solution $(\boldsymbol{x}, \boldsymbol{y})$, for each O/D pair $w \in W$ there could be several paths with length not larger than u^w. Then, the values of the flow variables f_a^w will describe one of them, which can contain subtours and/or not be necessarily the shortest one, but the path choice has no influence on the objective function values* (2.12) *and* (2.20)*, respectively.*

Remark 3. *For (MC) and (PC), since the objective is to design a network, it can be believed that they can be decomposed into two separate problems: first, designing a network and then channeling the flow through the network constructed. This interpretation is not correct. A suboptimal solution could be considered for either one, or even an infeasible solution in the case of (PC).*

Remark 4. *In the previous works by Marín and Jaramillo (2009) and García-Archilla et al. (2013), constraints* (2.16) *and* (2.17) *are formulated in a different way. For example, in García-Archilla et al. (2013), these constraints were written as*

$$f_a^w + z^w - 1 \leq x_a, \qquad w \in W,\ e = \{i,j\} \in E^w : a = (i,j), \tag{2.25}$$

$$\sum_{a \in A^w} d_a f_a^w + M(z^w - 1) \leq u^w z^w, \qquad w \in W, \tag{2.26}$$

where the design variable x_a is defined for each arc. Given that $z^w - 1 \leq 0$, expressions (2.16) *and* (2.17) *are stronger than* (2.25) *and* (2.26)*, respectively.*

In addition, constraint (2.26) *involves a "big-M" constant. Our proposed formulation does not need it, which avoids the numerical instability generated by this constant. As we will see in Subsection 2.4.2, we observed that our proposed formulation is not only stronger than the one proposed in García-Archilla et al. (2013), but it is also computationally more efficient. In consequence, we only focus our analysis on our proposed formulation.*

Remark 5. *Another observation is that constraints* (2.16) *are a reinforcement of the usual disaggregated capacity constraints:*

$$\begin{cases} f_a^w \leq x_e, \\ f_{\hat{a}}^w \leq x_e. \end{cases} \tag{2.27}$$

In most applications where flow or design variables appear in the objective functions, the disaggregated version is sufficient to obtain a valid model since subtours are naturally non-optimal and the optimal solution is composed by a set of shortest paths. However, it is not the case in our models, and there exist optimal solutions with subtours although the aggregated version (2.16) *is used, as explained in Remark 2. Such constraint avoid subtours of length two. The use of* (2.16) *instead of* (2.27) *can be simply taken as a reinforcement to the formulations to reduce the combinatorial part without increasing the number of constraints to the problems. Such a strengthening was already introduced in the context of Uncapacitated Network Design, see e.g. Balakrishnan et al. (1989), and Steiner Trees, see e.g. Sinnl and Ljubić (2016) and Fortz et al. (2021).*

2.2.2 Preprocessing methods

In this section we describe some methods to reduce the size of the instances before solving them. First, we describe how to build each subgraph $\mathcal{N}^w = (N^w, E^w)$, $w \in W$. Then for each problem, (MC) and (PC), we sketch a method to eliminate O/D pairs which will never be covered.

To create each subgraph $\mathcal{N}^w$ we only consider useful nodes and edges from $\mathcal{N}$. That is, we eliminate all the nodes $i \in N$ that do not belong to any path from w^s to w^t shorter than u^w. The resulting set is N^w. Then, we define E^w as the set of edges in E incident only to the non eliminated nodes. That is, N^w and E^w are the sets formed by all feasible nodes and edges for w. Finally, the set A^w is obtained by duplicating all edges in E^w with the exception of arcs (i, w^s) and (w^t, i). We describe this procedure in Algorithm 1.

We assume that the cost of constructing each node and each edge is not higher than the budget.

Algorithm 1: Preprocessing I

```
for $w \in W$ do
    $N^w = N$
    for $i \in N$ do
        compute the shortest path for the O/D pairs $(w^s, i)$ and $(i, w^t)$.
        if the sum of the length of both paths is greater than $u^w$ then
            $N^w = N^w \setminus \{i\}$
            $E^w = E^w \setminus \delta(i)$
        end
    end
    $A^w = \{(i,j) \in A : \{i,j\} \in E^w,\ j \neq w^s,\ i \neq w^t\}$
end
return $\{\mathcal{N}^w = (N^w, E^w), A^w\}_{w \in W}$
```

Next, we focus on (MC). We can eliminate O/D pairs w that are too expensive to be covered. That means, the O/D pair w is deleted from W if there is no path between w^s and w^t satisfying: i) its building cost is less than $\alpha\, C_{total}$; and ii) its length is less than u^w. This can be checked by solving a Shortest Path problem with Resource Constraints and can thus be done in a pseudo-polynomial time. Desrochers (1986) shows how to adapt Bellman-Ford algorithm to solve it. In Feillet et al. (2004) authors adapt this idea and strengthen it to solve the Elementary Shortest Path problem with Resource Constraints. However, given the moderate size of graphs we consider, we solve it as a feasibility problem. For each w, we consider the feasibility problem associated to constraints (2.13) (2.14), (2.15), (2.16), (2.17) and (2.18), with z^w fixed to 1. If this problem is infeasible, then the O/D pair w is deleted from W. Otherwise, there exists, at least, a *feasible path* denoted by Path^w. Pair w is a *feasible pair*, in terms coverage. We denote by $(\widetilde{N}^w, \widetilde{E}^w)$ the subgraph of $\mathcal{N}^w$ induced by Path^w.

2.2.3 Polyhedral properties

Both formulations (MC) and (PC) involve flow variables f_a^w whose number can be huge when the number of O/D pairs is large. To circumvent this drawback we use a Benders decomposition approach for solving (MC) and (PC). In this subsection, we present properties of the two formulations that allow us to apply such a decomposition in an efficient way.

The first proposition shows that we can relax the integrality constraints on the flow variables f_a^w. Let (MC_R) and (PC_R) denote the formulations (MC) and (PC) in which constraints (2.19) are replaced by non-negativity constraints, i.e.

$$f_a^w \geq 0, w \in W, a \in A. \tag{2.28}$$

We denote the set of feasible points to a formulation F by $\mathcal{F}(F)$. Regarding the notion of *projection* of a set of points (see Definition 1), the following statement is proved, after executing the previous preprocessing methods.

Proposition 2. *The projections of* (MC) *and* (MC_R) *onto the* $\boldsymbol{f}$*-space coincide. The same is true for* (PC)*. That is,*

$$Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC)) = Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC\_R))$$

and

$$Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(PC)) = Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(PC\_R)).$$

Proof. We provide the proof for (MC), the other one being identical.
First, $\mathcal{F}(MC) \subseteq \mathcal{F}(MC\_R)$ implies $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC)) \subseteq Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC\_R))$. Second, let $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ be a point belonging to $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC\_R))$. For every O/D pair $w \in W$ such that $z^w = 0$ then $\boldsymbol{f}^w = 0$. In the case where $z^w = 1$, there exists a flow $f_a^w \geq 0$ satisfying (2.15) and (2.16) that can be decomposed into a convex combination of flows on paths from w^s to w^t and cycles. Given that the flow f_a^w also satisfies (2.17), then a flow of value 1 on one of the paths in the convex combination must satisfy this constraint. Hence, by taking f_a^w equal to 1 for the arcs belonging to this path and to 0 otherwise, we show that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ also belongs to $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC))$. $\square$

Note that a similar result is presented in the article Ljubić et al. (2019). Based on Proposition 2, we propose a Benders decomposition where variables f_a^w are projected out from the model and replaced by Benders feasibility cuts. As we will see in Section 2.3.3, we also consider the *Benders facet-defining cuts* proposed in Conforti and Wolsey (2019). To apply this technique it is necessary to get an interior point of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC\_R))$ (resp. $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(PC\_R)))$. The following property gives us an algorithmic tool to apply this technique to (MC).

Proposition 3. *After preprocessing, the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC\_R))$ is full-dimensional.*

Proof. To prove the result, we exhibit $|N|+|E|+|W|+1$ affinely independent feasible points:

- The 0 vector is feasible.
- For each $i \in N$, the points:

$$y_i = 1,\, y_{i'} = 0,\, i' \in N \setminus \{i\}, \quad x_e = 0,\, e \in E, \quad z^w = 0,\, w \in W.$$

- For each $e = \{i, j\} \in E$, the points:

$$y_k = 1,\, k \in e,\, y_k = 0,\, k \in N \setminus \{i, j\}, \quad x_e = 1,\, x_{e'} = 0,\, e' \in E \setminus \{e\},$$
$$z^w = 0,\, w \in W.$$

- For each $w \in W$, the points:

$$y_i = 1,\, i \in \widetilde{N}^w,\, y_i = 0,\, i \in N \setminus \widetilde{N}^w, \quad x_e = 1,\, e \in \widetilde{E}^w,\, x_e = 0,\, e \in E \setminus \widetilde{E}^w,$$
$$z^w = 1,\, z^{w'} = 0,\, w' \in W \setminus \{w\}.$$

Clearly these points are feasible and affinely independent. Thus, the polytope is full-dimensional. □

The proof of Proposition 3 gives us a way to compute an interior point of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(MC\_R))$. The average of these $|N|+|E|+|W|+1$ points is indeed such an interior point.

This is not the case for (PC), as we show in Example 1.

Example 1. Consider the instance of (PC) given by the data presented in Table 2.1 and Figure 2.1. We consider the case where at least half of the population must be covered, that is $\beta = 0.5$. In order to satisfy the trip coverage constraint (2.21), the O/D pair $w = (1, 4)$ must be covered. Hence $z^{(1,4)} = 1$ is an implicit equality. Furthermore, the only path with a length less than or equal to $u^{(1,4)} = 15$ is composed of edges {1,2} and {2,4}. Hence, $x_{\{1,2\}}$, $x_{\{2,4\}}$, y_1, y_2 and y_4 must take value 1. As a consequence, the polytope associated to (PC) is not full-dimensional.

Origin	Destination	u^w	g^w
1	4	15	200
2	4	10	50
3	4	15	50

Table 2.1: Data in Example 1. We consider $\beta = 0.5$.

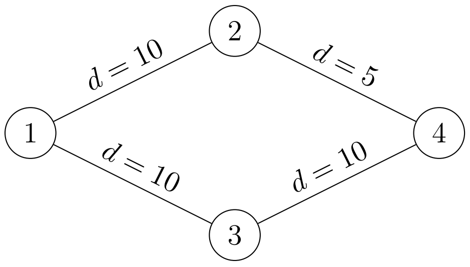


Figure 2.1: Graph of Example 1.

We can compute the dimension of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(PC\_R))$ in an algorithmic fashion. For that, we find feasible affinely independent points and, at the same time, we detect O/D pairs which must be covered in any feasible solution. Due to the latter, there are a subset of nodes and a subset of edges that have to be built in any feasible solution. This means that there is a subset of design variables $y_i, i \in N$, $x_e, e \in E$ and mode choice variables $z^w, w \in W$ that must be fixed to value 1. In this case, we say that O/D pair w is *essential*, and, if applicable, that edge e and node i are *essential for* w. At the opposite to (MC), a solution to (PC) with all variables set to 0 is not feasible. However, the solution obtained by serving all O/D pairs and building all nodes and edges is feasible. Therefore, we start with a solution with all variables in $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$ set to 1 and we check, one by one, if it is feasible to set them to 0. By setting one variable x_e or y_i to 0, it may become impossible to cover some essential O/D pair w. To simplify the notation, we introduce the binary parameters θ_e^w (and θ_i^w) taking value 1 if edge e (respectively node i) is essential for w. These new points are stored in a set S. Each time that the algorithm finds a variable that cannot be set to 0, we store it in sets $\bar{N}$, $\bar{E}$, $\bar{W}$, respectively. At the end of the algorithm, the dimension of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(PC\_R))$ is

$$\dim(\mathcal{P}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}) = |N| + |E| + |W| - \left(|\bar{N}| + |\bar{E}| + |\bar{W}|\right).$$

This procedure is depicted in Algorithm 2.

Algorithm 2 allows : i) to set some binary variables equal to 1, decreasing the problem size; and ii) to compute a relative interior point of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}(\mathcal{F}(PC\_R))$, necessary for the *facet-defining cuts*, as explained below in Section 2.3.3. The relative interior point is given by the average of the points in set S.

Algorithm 2: Computing the dimension for the polytope of (PC).

```
**Initialization:** Set $\bar{N} = \emptyset$, $\bar{E} = \emptyset$, $\bar{W} = \emptyset$ and $S = \emptyset$.
Add to set $S$: $(y_i = 1, i \in N, \quad x_e = 1, e \in E, \quad z^w = 1, w \in W)$
**for** $w' \in W$ **do**
    **if** $\sum_{w \in W \setminus \{w'\}} g^w \geq \beta\, G_{total}$ **then**
        Add to set $S$:
        $\Big(y_i = 1, i \in N, \quad x_e = 1, e \in E, \quad z^{w'} = 0,\ z^w = 1, w \in W \setminus \{w'\}\Big)$
    **else**
        $\bar{W} = \bar{W} \cup \{w'\}$.
    **end**
    **for** $e = \{i,j\} \in E$ **do**
        Compute shortest path from $w'^s$ to $w'^t$ in the graph $(N^{w'}, E^{w'} \setminus \{e\})$.
        **if** *the length of the shortest path is greater than* $u^{w'}$ ***or*** *there is no*
         *path between* $w'^s$ *and* $w'^t$ **then**
            $\bar{N} = \bar{N} \cup \{i,j\}$
            $\bar{E} = \bar{E} \cup \{e\}$
        **end**
    **end**
**end**
**for** $e' \in E \setminus \bar{E}$ **do**
    Add to set $S$:
    $\big(y_i = 1,\ i \in N, \quad x_e = 1, e \in E \setminus \{e'\}, x_{e'} = 0, \quad z^w = 1 - \theta^w_{e'}, w \in W\big)$
**end**
**for** $i' \in N \setminus \bar{N}$ **do**
    Add to set $S$:
    $(y_{i'} = 0,\ y_i = 1, i \in N \setminus \{i'\}, \quad x_e = 0, i' \in e,\ x_e = 1, i' \notin e,$
     $z^w = 1 - \theta^w_{i'}, w \in W\big)$
**end**
$\dim(\mathcal{P}_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}) = |N| + |E| + |W| - \big(|\bar{N}| + |\bar{E}| + |\bar{W}|\big)$
**return** $\bar{N}$, $\bar{E}$, $\bar{W}$, $S$ and $\dim(conv(P_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}))$
```

Example 1 cont. Regarding the previous example and following Algorithm 2, the O/D pair $(1,4)$ must be covered, $z^{(1,4)} = 1$. It is essential. Due to that, as its shortest path in the networks $(N^{(1,4)}, E^{(1,4)} \setminus \{\{1,2\}\})$ and $(N^{(1,4)}, E^{(1,4)} \setminus \{\{2,4\}\})$ is greater than $u^{(1,4)} = 15$, variables $x_{\{1,2\}}$, $x_{\{2,4\}}$, y_1, y_2, y_4 are set to 1. These edges are essential for it. Finally, the dimension of this polyhedron is

$$\dim(P_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}) = 4 + 4 + 3 - (3 + 2 + 1) = 5.$$

The relative interior point computed is:

$$x_{\{1,2\}} = 1,\ x_{\{2,4\}} = 1, x_{\{1,3\}} = \frac{5}{6},\ x_{\{3,4\}} = \frac{2}{3},\ y_1 = 1,\ y_2 = 1,\ y_3 = \frac{5}{6},\ y_4 = \frac{5}{6},$$
$$z^{(1,4)} = 1,\ z^{(2,4)} = \frac{5}{6},\ z^{(3,4)} = \frac{1}{2}.$$

2.2.4 Setting an initial solution

We determine an initial feasible solution for (MC) and (PC) with a simple greedy heuristic in which we sequentially select O/D pairs with best ratio demand over-building cost. More precisely, given the potential network $\mathcal{N} = (N, E)$, we compute for each O/D pair w the ratio $r^w = \frac{g^w}{C(\text{Path}^w)}$, where $C(\text{Path}^w)$ is the cost of a feasible path for w. We order these ratios decreasingly. We use this initial order in the heuristic for both (MC) and (PC). For (MC) the method proceeds as follows. It starts with an empty list of built nodes and edges, an empty list of O/D pairs covered, and a total cost set to 0. For each O/D pair w, in decreasing order of r^w, the heuristic tries to build Path^w considering edges and nodes that are already built. If the additional cost plus the current cost is less than the budget $\alpha\, C_{total}$, nodes and edges in Path^w are built and the O/D pair w is covered (i.e. $z^w = 1$). The total cost, the lists of built nodes and edges are updated. Otherwise we proceed with the next O/D pair. At the end of the algorithm we have an initial feasible solution.

To get an initial solution for (PC) we start with a list of all the O/D pairs covered and the amount of population covered equal to G_{total}. For each O/D pair w, in decreasing order of r^w, the algorithm checks if by deleting the O/D pair w from the list, the coverage constraint (2.21) is satisfied. If so, the O/D pair w is deleted from the list and the amount of population covered is updated. Finally, the algorithm builds the union of the subgraphs $(\widetilde{N}^w, \widetilde{E}^w)$ induced by the paths $\widetilde{l}^w$ for all the O/D pairs covered. Note that both initial solutions can be computed by solving $|W|$ shortest paths problems. These tasks can be executed much faster than solving (MC) and (PC) to optimality.

Pseudo-codes for both routines are provided in Appendix A. In Section 2.4, we will show the efficiency of adding this initial solution at the beginning of the branch-and-Benders-cut procedure.

2.2.5 Relation between (MC) and (PC)

With respect to the (MC) problem, there exists a set of optimal solution networks for which the demand satisfied is the same, but the cost of these optimal solution networks does not have to be necessarily the same. Then, with the aim of designing the cheapest network which maximizes the demand covered, once the (MC) is solved, its objective value is used to set the lower bound for the coverage constraint (2.21) of (PC) problem and solve it. In this manner, the (PC) problem builds a network that covers the same demand percentage as the (MC), but it could be at a lower cost. This situation is shown with the following example.

Example 2. Consider the instance given by the data presented in Table 2.2 and Figure 2.2 for the case in which the available budget is upper bounded by the 67% of the total cost of the underlying network, that is, $0.67\,C_{total} = 40$. We observe that the (MC) problem for this given instance has two optimal solution networks $\mathcal{N}_1 = (\{1,2,3\}, \{\{1,2\},\{1,3\}\})$ and $\mathcal{N}_2 = (\{2,4\}, \{\{2,4\}\})$ whose objective values are equal to 200 and the construction cost is 40 and 25, respectively. If the objective is to build the cheapest network which maximizes the demand covered, once the (MC) problem is solved, the (PC) problem must be computed with the lower bound of the trip coverage constraint $\beta = 0.5$, to cover the same percentage as in the optimal solution network for (MC). In this way, $\mathcal{N}_2$ is the cheapest optimal solution network for (MC) problem. Besides, $\mathcal{N}_2$ is the optimal solution for (PC) problem if $\beta = 0.5$.

Origin	Destination	u^w	g^w
1	2	15	50
2	3	30	150
2	4	15	200

Table 2.2: Data in Example 2.

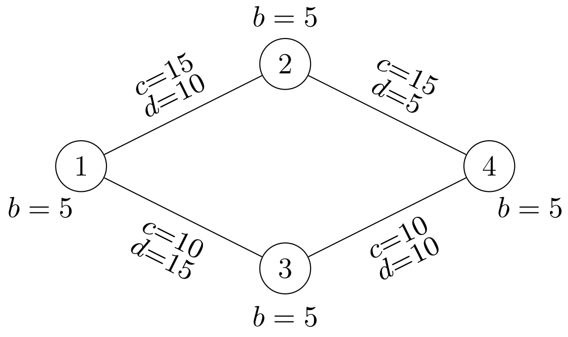


Figure 2.2: Graph of Example 2.
We consider $\alpha = 0.67$.

2.3 Benders decomposition implementations

In the following, we describe different Benders implementations for (MC) and (PC) obtained by projecting out variables f_a^w. Given that (MC) and (PC) share the same *subproblem* structure, the Benders decomposition applied to (MC) is valid for (PC) and vice versa. Thus, we will apply the same Benders decomposition for both problems throughout this chapter. These implementations are used as subroutines in a branch-and-Benders-cut scheme. This scheme allows cutting infeasible solutions along the *branch-and-bound* tree. Depending on the implementation, infeasible solutions can be separated at any node in the branch-and-bound tree or only when an integer solution is found. In the case of (MC), the *master problem* that we solve is:

$$
\begin{aligned}
(M\_MC) \quad & \max_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} \quad \sum_{w \in W} g^w z^w && (2.29)\\
& \text{s.t.} \quad (2.13), \ \ (2.14), \ \ (2.18)\\
& \qquad + \{\text{Benders Cuts } (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})\}.
\end{aligned}
$$

The master problem for (PC), named (M_PC), is stated analogously.

In Section 2.3.1, we discuss the *standard Benders cuts* obtained by dualizing the respective feasibility subproblem. Then, in Section 2.3.2 we discuss ways of generating normalized subproblems, to produce stronger cuts. We name them *normalized Benders cuts*. In Section 2.3.3, we apply *facet-defining Benders cuts* in order to get stronger cuts too, as proposed in Conforti and Wolsey (2019). Finally, we discuss an implementation where, at the beginning, *cut-set inequalities* are added to enhance the link between $\boldsymbol{x}$ and $\boldsymbol{z}$, and then Benders cuts are added.

2.3.1 LP feasibility cuts

Since the structure of the model allows it, we consider a *feasibility subproblem* made of constraints (2.15), (2.16), (2.17) and (2.28) for each commodity $w \in W$, denoted by $(SP)^w$. As it is clear from the context, we remove the index w from the notation. The dual of each feasibility subproblem can be expressed as:

$$
\begin{aligned}
(DSP)^w \quad & \max_{\boldsymbol{\gamma},\boldsymbol{\sigma},\boldsymbol{\upsilon}} \quad z\,\gamma_{w^s} - \sum_{e \in E} x_e\,\sigma_e - u\,z\,\upsilon && (2.30)\\
& \text{s.t.} \quad \gamma_i - \gamma_j - \sigma_e - d_a\,\upsilon \le 0, \qquad a = (i,j) \in A : e = \{i,j\}, && (2.31)\\
& \qquad \sigma_e,\ \upsilon \ge 0, \qquad e \in E, && (2.32)
\end{aligned}
$$

where $\boldsymbol{\gamma}$ is the vector of dual variables related to constraints (2.15), $\boldsymbol{\sigma}$ is the vector of dual variables corresponding to the set of constraints (2.16) and $\boldsymbol{\upsilon}$ is the dual variable of constraint (2.17).

Remark 6. *Since constraints* (2.15) *are linearly dependent, we set* $\gamma_{w^t} = 0$.

Given a solution $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ of the master problem, there are two possible outcomes for $(SP)^w$:

1. $(SP)^w$ is infeasible and $(DSP)^w$ is unbounded. Then, there exists an increasing direction $(\boldsymbol{\gamma}, \boldsymbol{\sigma}, \boldsymbol{\upsilon})$ with positive cost. In this case, the current solution $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ is cut by
$$(\gamma_{w^s} - u\,\upsilon)\,z - \sum_{e \in E} \sigma_e\,x_e \le 0. \tag{2.33}$$
2. $(SP)^w$ is feasible and consequently, $(DSP)^w$ has an optimal objective value equal to zero. In this case, no cut is added.

Remark 7. *We note that each subproblem* $(SP)^w$ *is feasible whenever* $z^w = 0$*, so it is necessary to check feasibility to add feasibility cuts* (2.33) *only in the case where* $z^w > 0$.

2.3.2 Normalized Benders cuts

The overall branch-and-Benders-cut performance heavily relies on how the cuts are implemented. It is known that feasibility cuts may have poor performance due to the lack of ability of selecting a *good* extreme ray (see, for example, Fischetti et al. (2010); Ljubić et al. (2012)). However, normalization techniques are known to be efficient to overcome this drawback (see Magnanti and Wong (1981); Balas and Perregaard (2002, 2003)). The main idea is to transform extreme rays in extreme points of a suitable polytope. In this section we study three ways to normalize the dual subproblem described above.

First, we note that the feasibility subproblem made of constraints (2.15), (2.16), (2.17) and (2.28) can be reformulated as a *min cost flow* problem in $\mathcal{N}^w$ with capacities $\boldsymbol{x}$ and arc costs $\boldsymbol{d}$, called as *normalized subproblem* (NSP).

$$
\begin{aligned}
(NSP)^w \quad & \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{f}} \quad \sum_{a\in A} d_a\, f_a && (2.34)\\
& \text{s.t.} \quad (2.15),\ (2.16),\ (2.28).
\end{aligned}
$$

The associated dual subproblem is:

$$
\begin{aligned}
(DNSP)^w \quad & \max_{\boldsymbol{\gamma},\boldsymbol{\sigma}} \quad z\,\alpha_{w^s} - \sum_{e\in E}\sigma_e x_e && && (2.35)\\
& \text{s.t.} \quad \gamma_i - \gamma_j - \sigma_e \le d_a, && a=(i,j)\in A : e=\{i,j\}, && (2.36)\\
& \qquad\quad \sigma_e \ge 0, && e\in E. && (2.37)
\end{aligned}
$$

Regarding Remark 7, as in $(SP)^w$, the primal subproblem $(NSP)^w$ may be infeasible whenever $z^w > 0$. Subproblems $(NSP)^w$ are no longer feasibility problems, although some of their respective dual forms can be unbounded. As the splitting demand constraint has to be satisfied there are two kind of cuts to add:

1. $(NSP)^w$ is infeasible and $(DNSP)^w$ is unbounded. In this case, the solution $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$ is cut by the constraint
$$\gamma_{w^s}\, z - \sum_{e\in E}\sigma_e\, x_e \le 0. \tag{2.38}$$
2. $(NSP)^w$ is feasible and $(DNSP)^w$ has optimal solution. Consequently, if their solutions $(\boldsymbol{\gamma},\boldsymbol{\sigma})$ and $(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z})$ satisfy that $\gamma_{w^s}\, z - \sum_{e\in E}\sigma_e\, x_e > u\, z$ then, the

solution $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ is discarded to the master problem by the constraint

$$(\gamma_{w^s} - u)\, z - \sum_{e \in E} \sigma_e \, x_e \leq 0. \tag{2.39}$$

We refer to this implementation as `BD_Norm1`.

In this situation, as it has already been seen, there still exists dual subproblems $(DNSP)^w$ which generate extreme rays. We refer to `BD_Norm2` as second dual normalization obtained by adding the dual constraint $\gamma_{w^s} = u + 1$ to each subproblem. In this case, every extreme ray of $(SP)^w$ corresponds to one of the extreme points of $(NSP)^w$. A cut is added whenever the optimal dual objective value is positive. This cut has the form

$$z - \sum_{e \in E} \sigma_e \, x_e \leq 0. \tag{2.40}$$

We finally tested a third dual normalization, `BD_Norm3`, by adding constraints

$$\sigma_e \leq 1, \qquad e \in E, \tag{2.41}$$

directly in $(DSP)^w$.

We tested the three dual normalizations described above for (MC) using randomly generated networks with 10, 20 and 40 nodes, as described in Subsection 2.4.1. As we will see in Subsection 2.4.2, only `BD_Norm1` results to be competitive.

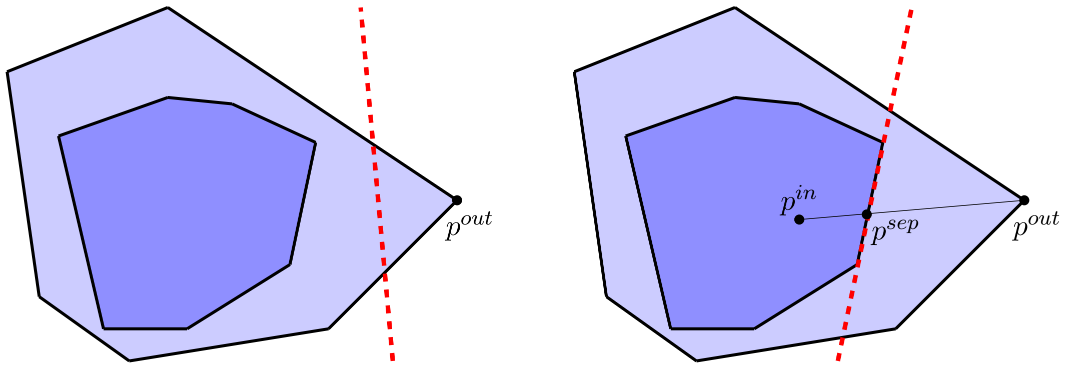
2.3.3 Facet-defining Benders cuts

Here we describe how to generate Benders cuts for (MC) based on the ideas exposed in Conforti and Wolsey (2019). The procedure for (PC) is the same. Given an *interior point* or *core point* $(\boldsymbol{x}^{in}, \boldsymbol{y}^{in}, \boldsymbol{z}^{in})$ of the convex hull of feasible solutions and an *exterior point* $(\boldsymbol{x}^{out}, \boldsymbol{y}^{out}, \boldsymbol{z}^{out})$, that is a solution of the LP relaxation of the current restricted master problem, a cut that induces a facet or an improper face of the polyhedron defined by the LP relaxation of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}\mathcal{F}(MC)$ is generated. We denote the difference $\boldsymbol{x}^{out} - \boldsymbol{x}^{in}$ by $\Delta\boldsymbol{x}$. We define $\Delta\boldsymbol{y}$ and $\Delta\boldsymbol{z}$ analogously. The idea is to find the furthest point from the core point, feasible to the LP-relaxation of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}}\mathcal{F}(MC)$ and lying on the segment line between the *core point* and the *exterior point*. This point is of the form $(\boldsymbol{x}^{sep}, \boldsymbol{y}^{sep}, \boldsymbol{z}^{sep}) = (\boldsymbol{x}^{out}, \boldsymbol{y}^{out}, \boldsymbol{z}^{out}) - \lambda(\Delta\boldsymbol{x}, \Delta\boldsymbol{y}, \Delta\boldsymbol{z})$. Figure 2.3b shows this type of cuts. We refer the problem of generating such facet-defining Benders cuts as *facet-defining subproblem*, (FSP), and it reads as follows:

$$
\begin{aligned}
(FSP)^w \quad & \min_{\boldsymbol{f},\lambda} \lambda && && (2.42)\\
\text{s.t.} \quad & \sum_{a\in\delta_+^w(i)} f_a - \sum_{a\in\delta_-^w(i)} f_a = \begin{cases} z^{out} - \lambda\,\Delta z, & \text{if } i = w^s,\\ 0, & \text{otherwise,}\end{cases} && && (2.43)\\
& f_a + f_{\hat{a}} \le x_e^{out} - \lambda\,\Delta x_e, \; e=\{i,j\}\in E : a=(i,j), \hat{a}=(j,i), && && (2.44)\\
& \sum_{a\in A} d_a\, f_a \le u\, z^{out} - u\,\Delta z\,\lambda, && && (2.45)\\
& 0 \le \lambda \le 1, && && (2.46)\\
& f_a \ge 0, && a\in A. && (2.47)
\end{aligned}
$$

In order to obtain the Benders feasibility cut we solve its associated dual:

$$
\begin{aligned}
(DFSP)^w \quad & \max_{\boldsymbol{\gamma},\boldsymbol{\sigma},\boldsymbol{\upsilon}} \quad z^{out}\,\gamma_{w^s} - \sum_{e\in E} x_e^{out}\,\sigma_e - u\,z^{out}\,\upsilon && && (2.48)\\
& \text{s.t.} \quad \Delta z\,\gamma_{w^s} - \sum_{e\in E} \Delta x_e\,\sigma_e - u\,\Delta z\,\upsilon \le 1, && && (2.49)\\
& \qquad \gamma_i - \gamma_j - \sigma_e - d_a\,\upsilon \le 0, && a=(i,j)\in A : e=\{i,j\},\\
& \qquad \sigma_e,\, \upsilon \ge 0, && e\in E.
\end{aligned}
$$



(a) Standard Benders cuts (b) Facet-defining Benders cuts

Figure 2.3: Benders cuts

Given that $(FSP)^w$ is always feasible ($\lambda = 1$ is feasible) and that its optimal value is lower bounded by 0, then, both $(FSP)^w$ and $(DFSP)^w$ have always finite optimal solutions. Whenever the optimal value of λ is 0, $(\boldsymbol{x}^{out}, \boldsymbol{y}^{out}, \boldsymbol{z}^{out})$ is feasible. A cut is added if the optimal value of $(DFSP)^w$ is strictly greater than 0. The new cut has the same form as in (2.33). Note that this problem can be seen as a dual normalized version of $(SP)^w$ with the dual constraint (2.49). This approach is an improvement in comparison with the stabilization cuts proposed by Ben-Ameur and Neto (2007), where λ is a fixed parameter.

Core points can be obtained by computing the average of the points described in the proof of Proposition 3 for (MC) formulation and the average of the points in list S obtained by applying Algorithm 2 for (PC) formulation.

2.3.4 Cut-set inequalities

By projecting out variables vector $\boldsymbol{f}$, information regarding the link between vectors $\boldsymbol{x}$ and $\boldsymbol{z}$ is lost. To deal with that, we use *cut-set inequalities*, which are a type of valid inequalities family widely used in different algorithms for Network Design problems. Some articles in which they have been studied are, for instance, Barahona (1996), Koster et al. (2013) and Costa et al. (2009). In our case, the cut-set inequalities represent the information lost with respect to the connectivity for the O/D pair w in the solution given by the design variables vector $\boldsymbol{x}$, as shown in Figure 2.4.

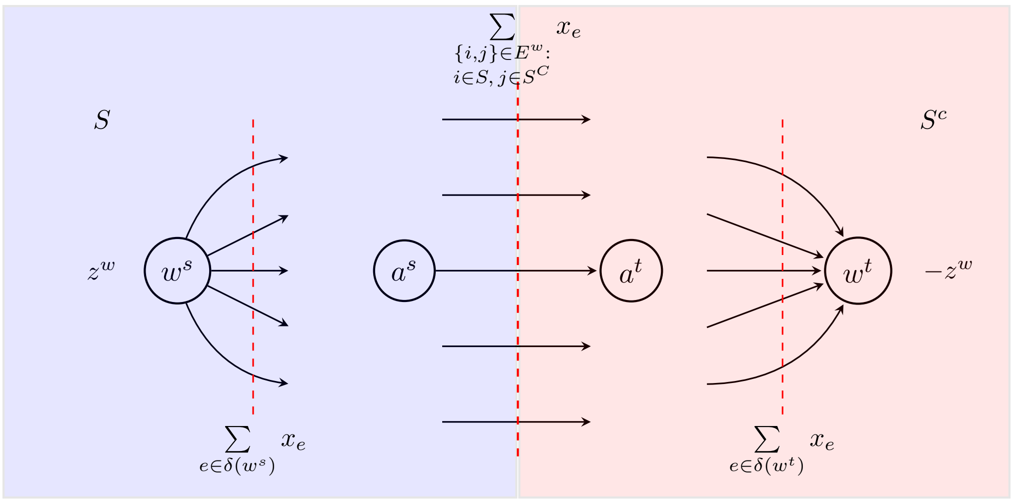

$\sum_{\substack{\{i,j\}\in E^w:\\ i\in S,\, j\in S^C}} x_e$
S
S^c
z^w
w^s
a^s
a^t
w^t
$-z^w$
$\sum_{e\in\delta(w^s)} x_e$
$\sum_{e\in\delta(w^t)} x_e$

Figure 2.4: Cut-set inequalities

Definition 2. *Let (S, S^C) be a (w^s, w^t)-partition of N^w for a fixed O/D pair w, i.e. (S, S^C) satisfies: i) $w^s \in S$; ii) $w^t \in S^C$, with $S^C = N \setminus S$ its complement. A cut-set inequality is defined as*

$$z^w \leq \sum_{\substack{\{i,j\}\in E^w:\\ i\in S,\, j\in S^C}} x_{\{i,j\}}, \quad w \in W: \quad (S, S^C) \text{ is a } (w^s, w^t)\text{-partition of } N^w. \tag{2.50}$$

It is easy to see that cut-set inequalities belong to the LP-based Benders family, as explained in Remark 8.

Remark 8. *Note that it is easy to see that cut-set inequalities belong to the LP-based Benders family 2.33. Let (S, S^C) be a (w^s, w^t)-partition in the graph $\mathcal{N}^w$ for $w \in W$. Consider the following dual solution:*

- $\gamma_i = 1$ *if* $i \in S$; $\gamma_i = 0$ *if* $i \in S^C$.
- $\sigma_e = 1$ *if* $e = \{i,j\} \in E^w$, $i \in S$, $j \in S^C$; $\sigma_e = 0$, *otherwise.*
- $\upsilon = 0$.

This solution is feasible to $(DSP)^w$ *and induces a cut as* (2.50).

In order to improve computational performance of the Benders procedure, we test two approaches to include these inequalities:

1. We implement a modification of the Benders callback algorithm with the following idea. First, for each $w \in W$, using the solution vector $(\boldsymbol{x}, \boldsymbol{y})$ from the master problem, the algorithm generates a network (N^w, E^w) with capacity 1 for each edge built. Then, a *Depth-First Search* algorithm is applied to obtain the connected component containing w^s. If the connected component does not contain w^t, a cut of the form (2.50) is added. Otherwise, we generate a Benders cut as before. This routine is depicted in Algorithm 3.

   Algorithm 3: Callback implementation with cut-set inequalities.

   ```
   Require: ($x_e, e \in E$, $y_i, i \in N$, $z^w, w \in W$) from the master solution vector ($\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$).
   for $w \in W$ do
       Build graph $(N^w(\boldsymbol{y}), E^w(\boldsymbol{x}))$ induced by the solution vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ from the master.
       Compute the connected component $S$ in $(N^w(\boldsymbol{x}), E^w(\boldsymbol{x}))$ containing $w^s$.
       if $w^t$ is not included in $S$ then
           Add the cut $z^w \le \sum_{\substack{\{i,j\} \in E^w: \\ i \in S, j \in S^C}} x_{\{i,j\}}$.
       else
           Solve the corresponding subproblem ($(DSP)^w$, $(DNSP)^w$, $(DFSP)^w$) and add cut if it is necessary.
       end
   end
   return Cut.
   ```

   We tested this implementation with subproblems $(DFSP)^w$. We observed that the convergence is slower and we generate more cuts. These preliminary results are shown in Table 2.6.

2. We add to the master problem the cut-set inequalities respective to the origin and at the destination of each O/D pair $w \in W$ at the beginning of the algorithm. These valid inequalities have the form

$$\begin{cases} z^w \leq \sum\limits_{e \in \delta(w^s)} x_e, \\ z^w \leq \sum\limits_{e \in \delta(w^t)} x_e. \end{cases} \tag{2.51}$$

This means that for each O/D pair to be covered, there should exist at least one edge incident to its origin and one edge incident to its destination, i.e. each O/D pair should have at least one arc going out of its origin and another one coming into its destination.

2.4 Computational experiments

In this section, we compare the performance of the different families of Benders cuts presented in Section 2.3 using the branch-and-Benders-cut algorithm (denoted as `B&BC`).

All our computational experiments were performed on a computer equipped with an Intel Core i5-7300 CPU processor, with 2.50 gigahertz 4-core, and 16 gigabytes of RAM. The operating system is 64-bit Windows 10. Codes were implemented in Python 3.8. These experiments have been carried out through `CPLEX 12.10` solver, named `CPLEX`, using its Python interface. `CPLEX` parameters were set to their default values and the models were optimized in a single threaded mode.

For that, `t` denotes the average value for solution times given in seconds, `gap` denotes the average of relative optimality gaps in percent (the relative percent difference between the best solution and the best bound obtained within the time limit), `LP gap` denotes the average of LP gaps in percent and `cuts` is the average of number of cuts generated.

2.4.1 Data sets: benchmark networks and random instances

We divide the tested instances into two groups: *benchmark instances* and *random instances*. Our benchmark instances are composed by the Seville city network (García-Archilla et al., 2013) and Sioux Falls city network (Hellman, 2013), named as `Seville` and `Sioux Falls` in what follows.

The `Seville` instance is composed partially by the real data given by the authors of García-Archilla et al. (2013). From this data, we have used the topology of the underlying network, cost and distance vector for the set of arcs and the demand matrix. This network is composed of 49 nodes and 119 edges. Originally, the set of O/D pairs W was formed by all possible ones ($49 \cdot 48 = 2352$). However, some entries in the demand matrix of this instance are equal to 0 and we thus exclude them from the analysis. Specifically, 630 pairs have zero demand, almost 27% of the whole set. We consider a private utility u equal to twice the shortest path length in the underlying network. Each node cost is generated according to a uniform distribution $\mathcal{U}(7, 13)$. The available budget has been fixed to 30% of the cost of building the whole underlying network and the minimum proportion of demand to be covered to $\beta = 0.5$.

For the `Sioux Falls` instance, the topology of the network is described by 24 nodes and 38 edges. Set W is also formed by all possible O/D pairs ($38 \cdot 37 = 1406$). The parameters have been chosen in the same manner as for the random instances.

We generate our random instances as follows. We consider planar networks with a set of n nodes, with $n \in \{10, 20, 40, 60\}$. Nodes are placed in a grid of n square cells, each one of 10 units per side. For each cell, a point is randomly generated close to the center of the cell. For each setting of nodes we consider a planar graph with its maximum number of edges, deleting each edge with probability 0.3. We replicated this procedure 10 times for each n, so that the number of nodes is the same while the number of edges may vary. Therefore, there are 40 different underlying networks. We name these instances as N10, N20, N40 and N60. We provide the average cycle availability, connectivity and density for random instances networks in Table 2.3. A couple of them are depicted in Figure 2.5.

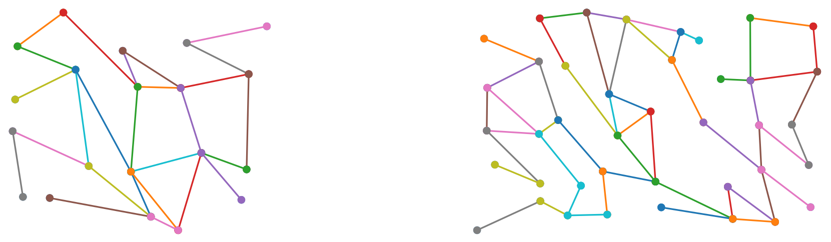


Figure 2.5: Example of underlying networks with $|N|$=20 and $|N|$=40.

Network	Cycle availability $\frac{\lvert E\rvert-\lvert N\rvert+1}{2\lvert N\rvert-5}$	Connectivity $\frac{\lvert E\rvert}{\lvert N\rvert}$	Density $\frac{\lvert E\rvert}{3(\lvert N\rvert-2)}$
N10	0.11	1.05	0.44
N20	0.11	1.12	0.41
N40	0.13	1.22	0.43
N60	0.16	1.29	0.45
Overall	0.12	1.17	0.43

Table 2.3: Cycle availability, connectivity and density parameters for the underlying networks in random instances.

Once a random instance $\mathcal{N}$ is generated, construction costs b_i, $i \in N$, are also randomly generated according to a uniform distribution $\mathcal{U}(7, 13)$. So, each node costs 10 monetary units on average. The construction cost of each edge $e \in E$, c_e, is set to its Euclidean length. This means that building the links cost 1 monetary unit per length unit. The node and edge costs are rounded to integer numbers. We set the budget αC_{total} equal to 50% of the cost of building the whole underlying network considered. That is, $\alpha = 0.5$.

To build set W, we randomly pick each possible O/D pair of nodes with probability 0.5. In consequence, this set has $\frac{n(n-1)}{2}$ pairs on average. Parameter u^w is set to 2 times the length of the shortest path between w^s and w^t in the potential network $\mathcal{N}$, $u^w = 2\, d_{\mathcal{N}}(w)$. Finally, the demand g^w for each O/D pair w is randomly generated according to the uniform distribution $\mathcal{U}(10, 300)$.

2.4.2 Preliminary experiments

Before presenting an extensive computational study of the algorithms, we provide some preliminary results to analyze: i) the efficiency of the formulation presented in García-Archilla et al. (2013); ii) the efficiency of the cut normalizations described in Subsection 2.3.2, and iii) the performance of the cut-set based Branch-and-cut procedure described in Subsection 2.3.4.

Network	`Formulation using` (2.16)-(2.17)		`Formulation using` (2.25)-(2.26)	
	`t`	`LP gap`	`t`	`LP gap`
N10	0.17	43.21	0.26	96.43
N20	5.78	56.33	228.22	106.71
	`gap`	`LP gap`	`gap`	`LP gap`
N40	11.74	68.15	54.85	137.13

Table 2.4: Comparing the performance of the two different types of mode choice and capacity constraints for (MC) within a time limit of 1 hour. The majority of N40 instances were not solved to optimality, then the average gap is shown.

We first show that our formulation using (2.16)-(2.17) is not only stronger than the one formulated with (2.25)-(2.26) but also more efficient. Table 2.4 shows some statistics for the two formulations discussed at the end of Subection 2.2.1, for instances with 10 and 20 nodes. We also tested instances with 40 nodes but most of them were not solved to optimality within one hour. In that case, we provide the optimality gap instead of the solution time. We consider 5 instances of each size. Note that constraints (2.26) are equivalent to constraints (2.17) by setting $M = 0$. We tested several positive values for M.

Secondly, we tested the three dual normalizations described in Subsection 2.3.2 for (MC). Table 2.5 shows average values obtained for solution time in seconds and number of cuts needed for this experiment. The only one that seems competitive is `BD_Norm1`. We observed that cut coefficients generated with `BD_Norm1` are mainly 0's or 1's. In the case of `BD_Norm2` and `BD_Norm3` we observe that coefficients generated are larger than the ones generated by `BD_Norm1`, so they may induce numerical instability. This situation is similar for the case of (PC).

Network	BD_Norm1		BD_Norm2		BD_Norm3	
	t	cuts	t	cuts	t	cuts
N10	0.21	44	0.22	47	0.24	104
N20	2.83	362	5.76	595	5.22	1418
N40	687.88	2904	*	*	*	*

Table 2.5: Comparing the performance of the three dual normalizations within a time limit of 1 hour for (MC). N10, N20 and N40 are refered to networks with 10, 20 and 40 nodes, respectively. The mark '*' indicates that four out of five instances were not solved within 1 hour.

Finally, we tested the cut-set inequalities implementation described in Subsection 2.3.4 with subproblems $(DFSP)^w$. We observed that convergence is slower and we generate more cuts. This might be due to the fact that these cuts do not include information about the length of the path in the graph, but only information regarding the existence of the path. These preliminary results are shown in Table 2.6, which provides average values obtained for solution times in seconds and the number of cuts added.

Network	BD_CW		Algorithm 3+BD_CW	
	t	cuts	t	cuts
N10	0.23	48	0.15	46
N20	2.47	411	2.53	500
N40	619.31	3486	722.02	3554

Table 2.6: Comparing the performance of the Algorithm 3 for (MC). N10, N20 and N40 refer to networks with 10, 20 and 40 nodes respectively.

In conclusion, all three implementations, with the exception of `BD_norm1`, are excluded from further analysis.

2.4.3 Branch-and-Benders-cuts performance

Our preliminary experiments show that including cuts only at integer nodes of the branch and bound tree is more efficient than including them in nodes with fractional solutions. Thus, in our experiments, we only separate integer solutions unless we specify the opposite. We used the `LazyConstraintCallback` function of `CPLEX` to separate integer solutions. Fractional solutions were separated using the `UserCutCallback` function. We study the different implementations of `B&BC` proposed in Subsections 2.3.1, 2.3.2 and 2.3.3. We use the following nomenclature:

- `BD_Trd`: `B&BC` algorithm using the feasibility Benders subproblems structure $(DSP)^w$, and its corresponding feasibility cuts (2.33).

- BD_Norm: B&BC algorithm using the normalized Benders subproblems structure $(DNSP)^w$, and its corresponding cuts (2.38) and (2.39).
- BD_CW: B&BC algorithm using the facet-defining Benders subproblems structure $(DFSP)^w$, and its corresponding feasibility cuts (2.33).

We compare our algorithms with the direct use of CPLEX, and the automatic Benders procedure proposed by CPLEX, noted by Auto_BD. CPLEX provides different implementations depending on the information that the user provides to the solver: i) CPLEX attempts to decompose the model strictly according to the decomposition provided by the user; ii) CPLEX decomposes the model by using this information as a hint and then refines the decomposition whenever possible; iii) CPLEX automatically decomposes the model, ignoring any information supplied by the user. We have tested these three possible settings, and only the first one is competitive.

Furthermore we have tested the following features:

- CS: If we include cut-set inequalities at each origin and destination as in (2.51).
- IS: If we provide an initial solution to the solver.
- RNC: If we add Benders cuts at the root node.

2.4.4 Branch-and-Benders-cuts performance on random instances

All the experiments have been performed with a limit of one hour of CPU time considering 10 instances of each size. Tables in this subsection show average values obtained for solution times in seconds, relative gaps in percent, and number of cuts needed. To determine these averages, we only consider the instances solved at optimality by all the algorithms.

First, we compare the performance of CPLEX for formulations (MC) and (PC) and the three different B&BC implementations described above (BD_Trd, BD_Norm and BD_CW). We also study the impact of the initial cut set inequalities CS in the efficiency of the proposed algorithms. Table 2.7 shows the performance of the algorithms for networks N10, N20 and N40. All the algorithms are able to solve at optimality N10 and N20 instances in less than 7 seconds for (MC) and (PC). For (MC) without CS, the fastest algorithm is BD_CW in sets N10, N20 and N40 for the instances solved to optimality. This is not the case for (PC), since we can observe that Auto_BD is slightly faster. In general, when CS is included, the solution time and the amount of cuts required decrease. Specifically, for (MC) in N40, the most efficient algorithm is BD_CW+CS which gets the optimal solution 43.8% faster than Auto_BD+CS. For (PC), it seems to be also profitable, since for N40 BD_CW+CS gets the optimal solution using 55% less time than Auto_BD. These results are shown in the second and fourth block of Table 2.7.

		Network	CPLEX	Auto_BD		BD_Trd		BD_Norm		BD_CW	
			t	t	cuts	t	cuts	t	cuts	t	cuts
(MC)	w.o. CS	N10	0.18	0.43	27	0.25	92	0.24	91	0.19	94
		N20	6.77	4.51	273	3.89	620	3.18	590	3.34	641
		N40	1646.93	617.85	1967	1095.25	3990	541.03	3677	457.81	4137
	+CS	N10	-	0.32	12	0.21	49	0.28	52	0.23	54
		N20	-	3.94	178	2.29	382	2.50	383	1.85	416
		N40	-	484.95	1248	637.49	2378	575.87	2530	272.39	3186
(PC)	w.o. CS	N10	0.18	0.29	16	0.24	92	0.28	89	0.20	91
		N20	6.73	4.87	305	3.55	607	4.68	681	2.15	606
		N40	2153.15	504.06	1752	657.59	4470	514.42	4246	837.41	4412
	+CS	N10	-	0.28	11	0.16	56	0.20	57	0.145	54
		N20	-	4.12	213	3.11	497	3.43	495	2.070	461
		N40	-	439.23	1527	261.74	3528	323.21	3583	197.55	3949

Table 2.7: Comparing the performance of the three Branch-and-Benders-cuts for (MC) and (PC).

Table 2.8 shows the instances in N40 solved in one hour. Without CS, some instances in set N40 cannot be solved to optimality neither for (MC) nor for (PC). Nevertheless, by including CS, Benders implementations can solve all the instances in N40 in the one hour limit.

		CPLEX	Auto_BD	BD_Trd	BD_Norm	BD_CW
(MC)	without CS	3	10	9	8	8
	+CS	-	10	10	10	10
(PC)	without CS	3	9	8	8	8
	+CS	-	10	10	10	10

Table 2.8: Instances N40 solved for (MC) and (PC) within a time limit of 1 hour.

We now concentrate on N60 instances. Table 2.9 compares the performance by adding cutset inequalities CS, setting an initial feasible solution IS and adding cuts at the root node RNC. We perform this experiment by computing the optimality gap after one hour. Without any of the features mentioned above, the trend on Table 2.7 is confirmed in (MC) for instances in set N60 where the optimality gap obtained after one hour is smaller in Auto_BD, see the first row in Table 2.9. However, for (PC) the gap after one hour is slightly better for BD_CW than for the other methods in this family (see the fifth row in Table 2.9). With respect to adding an initial solution, we observe that for (MC) it is only profitable for BD_CW+CS, obtaining on average a 3.5% better optimality gap than without it. The impact of adding an initial solution for (PC) is significant for BD_Trd+CS, BD_Norm+CS and BD_CW+CS obtaining on average solutions with a gap around 4% smaller. However, this improvement is not significant for Auto_BD for (PC) (see third row of both blocks in Table 2.9). Besides, we note that we obtain worse solutions by adding also RNC in both problems with all the algorithms tested. In summary, for the set of instances N60 we have that the best algorithm is `BD_CW+CS+IS` for (MC). It decreases the solution gap by around 8% comparing with the best option of `Auto_BD`, which is `Auto_BD+CS`. With regard to (PC), the best options are `BD_CW+CS+IS` and `BD_Norm+CS+IS`, since their solution gaps are around 5.5% smaller than the ones returned by `Auto_BD+CS`.

		`Auto_BD`		`BD_Trd`		`BD_Norm`		`BD_CW`	
		gap	cuts	gap	cuts	gap	cuts	gap	cuts
(MC)	w.o.{CS, IS, RNC}	38.54	6545	45.68	14068	44.53	13340	43.77	16707
	+CS	30.06	3729	24.27	8754	22.17	8912	25.76	11378
	+CS+IS	32.90	4987	27.23	9038	26.94	9469	22.27	11151
	+CS+IS+RNC	-		37.88	8054	37.92	8230	33.58	10834
(PC)	w.o.{CS, IS, RNC}	20.49	7009	20.40	14784	21.41	15501	19.93	15116
	+CS	15.92	5109	14.89	12354	14.09	11687	14.50	11744
	+CS+IS	15.86	4372	11.06	8961	10.47	8490	10.44	9683
	+CS+IS+RNC	-		20.93	10971	21.28	11449	19.94	11053

Table 2.9: Computing gaps to solve (MC) and (PC) comparing the performance of three families of Benders cuts. Instances N60 are used.

In the following, we analyze the performance of algorithms `BD_Norm+CS` `BD_CW+CS` when changing parameters α, β and u in the corresponding models. In Tables 2.10 and 2.11, we report average solution times and number of cuts needed to obtain optimal solutions for N40 for different values of these parameters. The instances are grouped by the three different increasing values of the available percentage of budget α (Table 2.10.a) or the minimum percentage of demand to cover β (Table 2.11.a) and private utility u (Tables 2.10.b and 2.11.b). For (MC), it is observed that the bigger the value of α is, the shorter the average solution time is. Table 2.10.b. shows that the larger the parameter u is, the shorter the solution time for `BD_Norm+CS` is. This behavior seems to be different if we are using `BD_CW+CS`.

α	`BD_Norm+CS`		`BD_CW+CS`	
	t	cuts	t	cuts
0.3	1053.56	1580	873.58	2017
0.5	622.45	2634	375.30	3358
0.7	151.24	3970	177.90	5035

a.

u	`BD_Norm+CS`		`BD_CW+CS`	
	t	cuts	t	cuts
$1.5\,d_{\mathcal{N}}$	802.05	2792	495.84	3041
$2\,d_{\mathcal{N}}$	622.46	2634	375.30	3358
$3\,d_{\mathcal{N}}$	591.02	2674	490.28	3173

b.

Table 2.10: Sensitivity analysis for (MC) with $|N| = 40$, using Benders cuts.

For (PC), Table 2.11.a shows that both algorithms take less time to solve the problem to optimality for $\beta = 0.7$ than for $\beta = 0.3$ and $\beta = 0.5$. `BD_CW+CS` is 5 minutes faster on average than `BD_Norm+CS` with $\beta = 0.5$. For $\beta = 0.3$ the result is the opposite, `BD_Norm+CS` is 100 seconds faster on average than `BD_CW+CS`. By varying u, we observe that the less the difference between public and private mode distances in the underlying network is, the longer it takes to reach optimality.

β	`BD_Norm+CS`		`BD_CW+CS`	
	`t`	`cuts`	`t`	`cuts`
0.3	640.28	2675	744.95	2848
0.5	697.87	3673	387.40	3914
0.7	273.53	3873	242.04	4460

a.

u	`BD_Norm+CS`		`BD_CW+CS`	
	`t`	`cuts`	`t`	`cuts`
$1.5\,d_{\mathcal{N}}$	653.47	3625	620.79	3613
$2\,d_{\mathcal{N}}$	697.87	3673	387.40	3914
$3\,d_{\mathcal{N}}$	561.43	3521	378.11	3643

b.

Table 2.11: Sensitivity analysis for (PC) with $|N| = 40$, using Benders cuts.

2.4.5 Branch-and-Benders-cuts performance on benchmark instances

We start by analyzing the `Seville` instance. Figures 2.6 and 2.7 show some results for this instance solved with `BD_CW+CS` for different parameter values. Points not connected in these graphs refer to those nodes that have not been built. The O/D pairs involving some of these nodes are thus not covered. They have been drawn to represent these not covered areas. Data corresponding to each case is collected at the bottom of its figure, in which `v(ILP)` refers to the objective value. For model (MC), parameter `cost` represents the cost of the network built, and, for (PC), G_{cov} makes reference to the demand covered. For (MC), we observe that smaller values of α carry larger solution times as in random instances. For (PC), as opposite to random instances, higher values of β are translated in larger solution times. Besides, in this instance, for both models, the shorter the parameter u is, the larger the solution times are.

α	u	GRASP			`BD_CW+CS`	
		`t`	`Best_Value`	`gap`	`t`	`v(ILP)`
0.2		110.829	48629	6.97	1036.11	52274
0.3	$2\,d_{\mathcal{N}}$	260.220	59828	3.96	313.07	62294
0.4		396.226	63546	0.72	21.36	64011
0.3	$1.5\,d_{\mathcal{N}}$	267.275	55778	6.97	2243.83	59958
	$3\,d_{\mathcal{N}}$	225.312	62049	0.99	113.88	62670

Table 2.12: Sensitivity analysis for GRASP of García-Archilla et al. (2013) for (MC) with `Seville` instance.

Furthermore, we compare the performance of the GRASP from García-Archilla et al. (2013), for which its main points are collected in Subsection 4.2, and our implementation `BD_CW+CS`. The goal of this experiment is to compare our implementation with a state-of-the-art heuristic for network design problems. We implemented the GRASP to run 5 times and return the best solution. Table 2.12 shows solution times, best value for GRASP (`Best_Value`), the optimality gap, and the optimal value computed with `BD_CW+CS` (`v(ILP)`). On the one hand, we observed that the more time `BD_CW+CS` takes to compute the optimal solution, the larger the gap of the solution returned by GRASP is. This happens for smaller values of the percentage α and utility u. On the other hand, for problems where GRASP obtains small optimality gap, BD_CW+CS is more efficient to compute the optimal solution. In other words, since GRASP is a constructive algorithm, it is not competitive for instances whose optimal solution captures most of the demand.

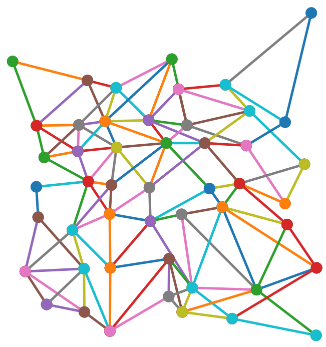
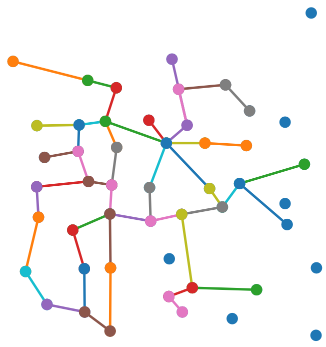
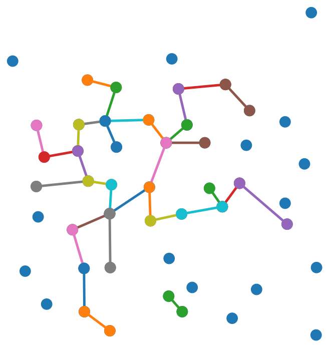
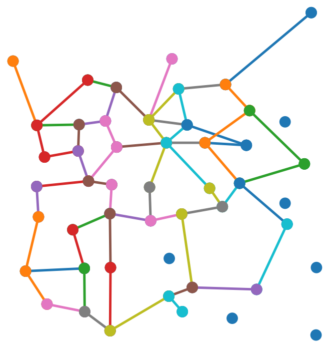
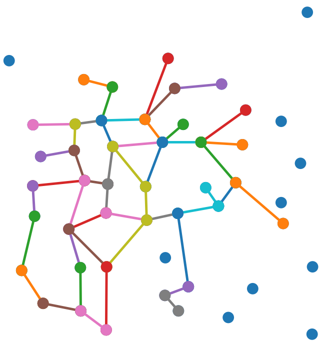
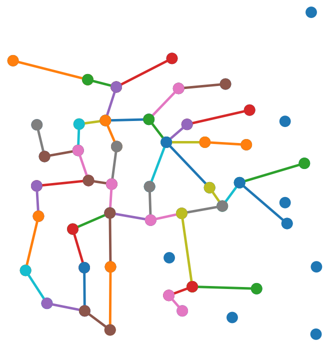
Underlying Network  $C_{total} = 295297826$, $G_{total} = 64011$	$\alpha = 0.3$, $u = 2\,d_{\mathcal{N}}$  t $= 313.07$, cuts $= 7149$ cost $= 88939.76$, v(ILP) $= 62294$
$\alpha = 0.2$, $u = 2\,d_{\mathcal{N}}$  t $= 1036.11$, cuts $= 8428$ cost $= 53362.53$, v(ILP) $= 52274$	$\alpha = 0.4$, $u = 2\,d_{\mathcal{N}}$  t $= 21.36$, cuts $= 2259$ cost $= 124353.84$, v(ILP) $= 64011$
$\alpha = 0.3$, $u = 1.5\,d_{\mathcal{N}}$  t$= 2243.83$, cuts $= 8537$ cost $= 88800.91$, v(ILP) $= 59958$	$\alpha = 0.3$, $u = 3\,d_{\mathcal{N}}$  t $= 113.88$, cuts $= 6400$ cost $= 88879.76$, v(ILP) $= 62670$

Figure 2.6: Sensitivity analysis for (MC) using Seville instance.

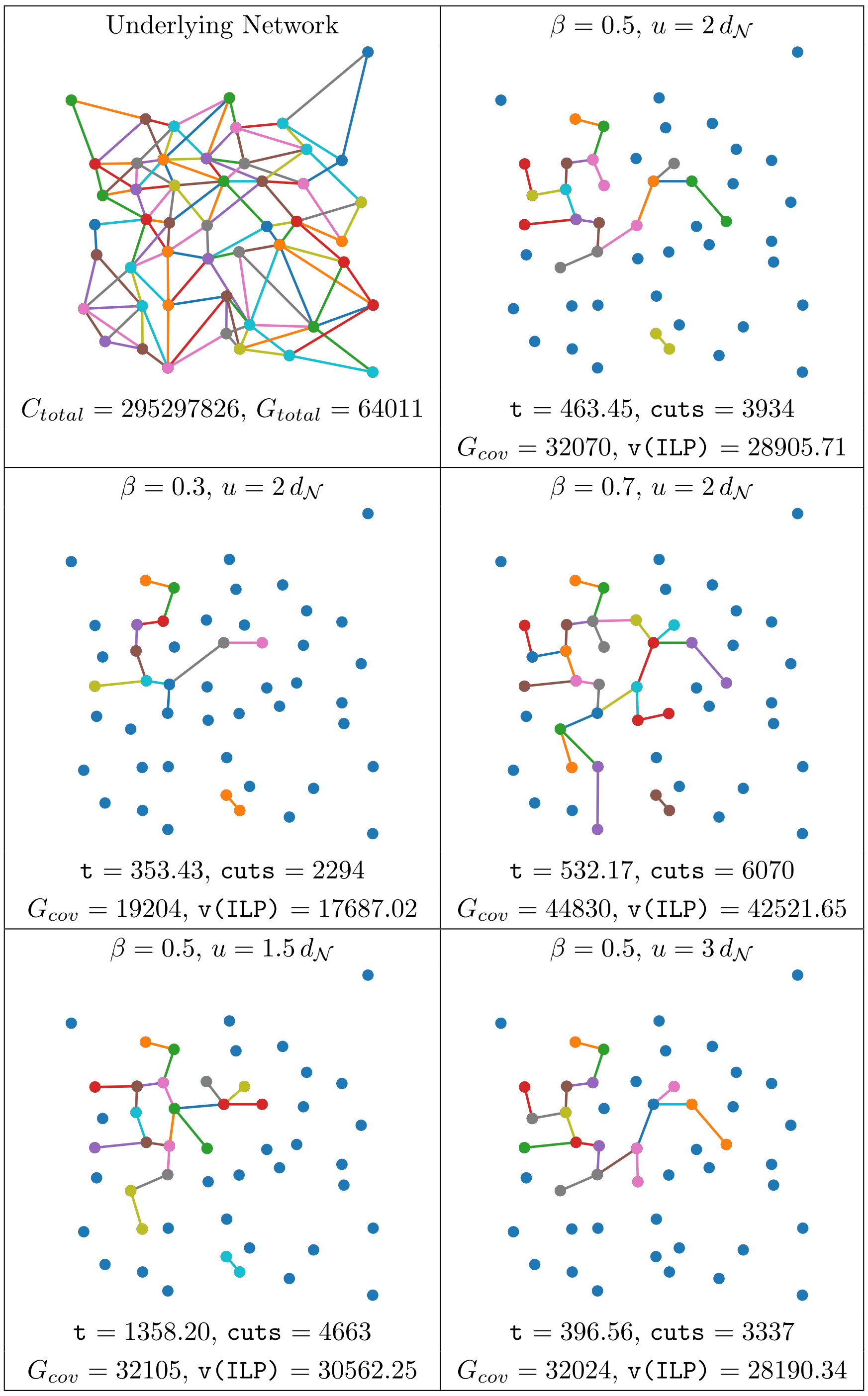

Underlying Network
$C_{total} = 295297826$, $G_{total} = 64011$
$\beta = 0.5$, $u = 2\,d_{\mathcal{N}}$
$\texttt{t} = 463.45$, $\texttt{cuts} = 3934$
$G_{cov} = 32070$, $\texttt{v(ILP)} = 28905.71$
$\beta = 0.3$, $u = 2\,d_{\mathcal{N}}$
$\texttt{t} = 353.43$, $\texttt{cuts} = 2294$
$G_{cov} = 19204$, $\texttt{v(ILP)} = 17687.02$
$\beta = 0.7$, $u = 2\,d_{\mathcal{N}}$
$\texttt{t} = 532.17$, $\texttt{cuts} = 6070$
$G_{cov} = 44830$, $\texttt{v(ILP)} = 42521.65$
$\beta = 0.5$, $u = 1.5\,d_{\mathcal{N}}$
$\texttt{t} = 1358.20$, $\texttt{cuts} = 4663$
$G_{cov} = 32105$, $\texttt{v(ILP)} = 30562.25$
$\beta = 0.5$, $u = 3\,d_{\mathcal{N}}$
$\texttt{t} = 396.56$, $\texttt{cuts} = 3337$
$G_{cov} = 32024$, $\texttt{v(ILP)} = 28190.34$

Figure 2.7: Sensitivity analysis for (PC) using `Seville` instance.

We discuss the results for the `Sioux Falls` instance, summarized in Figures B.1 and B.2 in Appendix B. We observe for (MC), as in the `Seville` network, that the smaller the values of α and u are, the larger the solution time is. The same is true when varying β in (PC), but not for u. It takes less time if the difference between both modes of transport is smaller or larger than $2\,d_{\mathcal{N}}$.

Our exact method is able to obtain the best quality solution, with a certificate of optimality in reasonable times. Given that network design problems are strategic decisions, having the best quality decision is often more important than the computational times. However, having efficient exact methods as the ones proposed in this chapter, allows decision makers to perform sensitivity analysis with optimality guarantees in reasonable times.

We also tested our algorithms on benchmark instances `Germany50` and `Ta2` from SNDLib (`http://sndlib.zib.de/`). We observed that adding cuts at the root node is beneficial for `Germany50`. We think that this behavior is due to the fact that `Germany50` has a denser potential graph (in particular, `Germany50` is not a planar graph). The rest of the results obtained for these instances are aligned with the results obtained for `Seville` and `Sioux Falls` instances. For the sake of shortness, this analysis is included in the supplementary material in `https://github.com/Natividad13/Network_Design_Instances`.

2.5 Conclusions

In this chapter, two variants of the Covering Network Design Problem have been studied: Maximal-Covering Network Design problem, which maximizes the demand covered under a budget constraint, and Partial-Covering Network Design problem, which minimizes the total constructing cost subject to a lower bound on the demand covered. For that, mixed-integer linear programming formulations are proposed, which are stronger than existing ones, for both problems. Besides, some polyhedral properties of these formulations are provided, useful from the algorithmic point of view. Then, we have developed exact methods based on Benders decomposition. We also discuss some preprocessing procedures to scale up the instances solved. These preprocessing techniques play a key role in order to obtain information about the instances and to derive a better algorithmic performance. Our computational results show that the techniques developed in this chapter allow obtaining better solutions in less time than the techniques in the existing literature. Further research on this topic is focused on the synergy of sophisticated heuristics to find good feasible solutions and decomposition methods, such as the ones presented in this chapter, to get better bounds and close the optimality gap. This further research has been started for the Maximal-Covering Network Design problem. In Chapter 4, three metaheuristic algorithms are presented and compared between them. Finally, we remark that objectives of (MC) and (PC) can be included in a bicriteria optimization model. An interesting extension is to exploit the decomposition methods described in this chapter to the multiobjective setting.

Chapter 3

Integrated Model with Rapid and Slow Transit Lines Design

Usually, when a rapid transit line is planned in a metropolitan area a slow transit system (e.g. bus system) already partially covers the demand of the new line. Thus, frequently the slow mode have to be cancelled or their routes modified.

In this chapter, we present an integrated formulation for designing a rapid transit line and re-locating an existing slow line with the aim of maximizing the demand covered. After that, we tackle the solution of the problem with a competitive Benders decomposition approach.

3.1 Introduction

The sequential problem of designing a rapid transit network and improving the feeder bus system

Re-routing bus lines is a very common task when a rapid transit line starts functioning. That is, usually, when a rapid transit line is planned, a slow transit mode/system (e.g. bus lines) already partially covers the demand of the new rapid transit line. Thus, when the rapid transit starts its regular services, the slow mode have to be cancelled or their routes modified to avoid becoming totally or partially useless. During the last five years, more than 80 new metro lines have been added to metro networks around the world and 27 new metro systems have been inaugurated. Therefore, more than one hundred new lines have become operational. Many other existing lines have been extended or upgraded. Moreover, new modern trams, train-trams, and commuter lines have also started their operation. In almost all the cases, bus lines were (partially) doing the service before. One typical example is the adaptation of the Bus Rapid Transit TranSantiago when Metro Line 7 started operating. Another example is the bus line 3 of TUSSAM (Urban Transport of Seville) with planned Line 3 of Metro of Seville. As a rule, the metro planning projects do not take into account the bus system because often they depend on different agencies.

The process of designing a rapid transit line and re-locating a slow line is planned according to a sequential method. Firstly, the rapid transit line is designed taking into account the private and the public flows, and possibly surveys on mobility in order to predict the future utilization of the new infrastructure and/or other criteria. Then, in a second stage, the bus route network is re-designed. However, this sequential process can lead to a suboptimal solution. The Feeder Buses Planning Problem has been researched to some extent (see Deng et al. (2013)). Models and algorithms for the Rapid Transit Network Design problem have been recently revised (see Laporte and Mesa (2020)). Nevertheless, as far as we are aware, for the design of a rapid transit line and the re-location of a slow line in an integrated manner no research has been done.

Structure of this chapter

The structure of the chapter is as follows. In Section 3.2, we present an integer mathematical programming formulation for the problem of designing simultaneously a rapid transit line and a new route of an existing slow line. Therefore, the aim is to design both lines with the objective of maximizing the number of passengers captured by both public modes against an existing private mode. Besides, some properties related to the number of transfer stations between both lines and a light preprocessing-routing are described. Furthermore, we solve the problem by a competitive Benders decomposition approach using the benchmark instance of `Seville` city network.

3.2 Problem formulation

In this section, we present a formulation that locates a Rapid Transit Line and Improves the Feeder Bus System in an integrated manner to maximize the demand to be captured by the set of both against an existing private mode. In this situation, a captured demand can combine both rapid and slow modes. That is, transfers are possible within the designed system. This mathematical formulation is named (RTL+$IFBS$) in what follows.

Data The formulation to be presented requires additional elements to those defined in Subsection 1.2.1, which we define below.

1. We consider the underlying network $\mathcal{N} = (N, E)$, used by the private mode, where N and E are the sets of nodes and edges. Without loss of generality, the rapid transit line to be located and the slow line to be re-located are subgraphs of $\mathcal{N}$ (see points 2 and 3 below).

2. The network $\mathcal{R} = (N_{\mathcal{R}}, E_{\mathcal{R}})$ is the subgraph of $\mathcal{N}$ from which the rapid transit line will be selected. Thus $N_{\mathcal{R}} \subset N$ and $E_{\mathcal{R}} \subset E$.

3. The network $\mathcal{S} = (N_{\mathcal{S}}, E_{\mathcal{S}})$ is the subgraph of $\mathcal{N}$ from which the slow line $\mathcal{S}$ will be selected. Thus $N_{\mathcal{S}} \subset N$ and $E_{\mathcal{S}} \subset E$.

4. For each $e = \{k, l\} \in E$, we define two arcs: $a = (k, l)$ and $\hat{a} = (k, l)$. The resulting set of arcs is denoted by A. With respect to each mode of transport we refer to the set of arcs by $A_{\mathcal{R}}$ and $A_{\mathcal{S}}$, respectively.

5. For the rapid transit line $\mathcal{R}$, there exists a maximum number of edges $E_{\mathcal{R}}^{max}$ to be constructed. For the slow line $\mathcal{S}$, bounds $E_{\mathcal{S}}^{max}$ and $E_{\mathcal{S}}^{id}$ are given to limit

the number of edges to be used and the minimum number of edges that must be coincident between the old and the modified line $\mathcal{S}$ (i.e. the number of edges not re-located). For that, vector $v_e^{\mathcal{S}}$, $e \in E_{\mathcal{S}}$ denotes the current path of $\mathcal{S}$.

6. For each mode of transport, we assume that there is a set of possible starting points, $O_{\mathcal{R}}$ and $O_{\mathcal{S}}$, of the lines. In the same way, sets containing possible end points $D_{\mathcal{R}}$ and $D_{\mathcal{S}}$.

7. Each O/D pair $w \in W$ has an associated subgraph $\mathcal{N}^w = (N^w, E^w) \subseteq \mathcal{N}$ containing all *feasible nodes* and *edges* for w, i.e. that belong to a path in $\mathcal{N}$ whose number of edges is smaller than or equal to $E_{\mathcal{R}}^{max}$ or $E_{\mathcal{S}}^{max}$, depending on the case (see the paragraph "Preprocessing methods" further in this chapter). We also denote A^w as the set of feasible arcs in $\mathcal{N}^w$.

8. For the neighborhood purpose, let $\delta(i)$ be the set of edges of $E_{\mathcal{R}}$ incident to node i. By $\delta^w(i)$ we denote the set of edges of $E_{\mathcal{R}}$ incident to node i in graph $\mathcal{N}^w$. Notation $\delta_+^w(i)$ ($\delta_-^w(i)$ respectively) is used to refer to the set of arcs of $A_{\mathcal{R}}$ going out (in, respectively) of node $i \in N^w$. In particular, we set $\delta_-^w(w^s) = \emptyset$ and $\delta_+^w(w^t) = \emptyset$. In the same way, we use notation $\vartheta^w(k)$ and $\vartheta_+^w(k)$ ($\vartheta_-^w(k)$ respectively) to denote the set of edges of $E_{\mathcal{S}}$ incident to node k and the set of arcs going out (in, respectively) of node k in $E_{\mathcal{S}}$. In particular, we set $\vartheta_-^w(w^s) = \emptyset$ and $\vartheta_+^w(w^t) = \emptyset$.

9. The set of possible transfer nodes is denoted by $N_{trans} \subseteq N_{\mathcal{R}} \cap N_{\mathcal{S}}$.

10. Other costs are those of traversing arc a in the rapid and slow mode, $t_a^{\mathcal{R}}$ and $t_a^{\mathcal{S}}$, respectively. The transfer cost at station k from $\mathcal{S}$ to $\mathcal{R}$ and from $\mathcal{R}$ to $\mathcal{S}$ are $t_k^{\mathcal{S}\mathcal{R}}$ and $t_k^{\mathcal{R}\mathcal{S}}$, respectively. The dwell time costs (stop times) are $t_{stop}^{\mathcal{R}}$ are $t_{stop}^{\mathcal{S}}$, which will be assumed independent from nodes. The waiting time at stations/stops, t_{wait}, is usually set as a half of the headway.

Variables According to the previous elements defined, the following binary variables are used:

1. $x_e^{\mathcal{R}} = 1$ if edge $e = \{k, l\} \in E_{\mathcal{R}}$ is included in the rapid public line $\mathcal{R}$; 0 otherwise. Analogously, $x_e^{\mathcal{S}} = 1$ if edge $e = \{k, l\} \in E_{\mathcal{S}}$ is included in the slow public line $\mathcal{S}$; 0 otherwise.

2. $y_i^{\mathcal{R}} = 1$ if node $i \in N$ is included in the alignment of the rapid system $\mathcal{R}$, but it does not stop on it; 0 otherwise.

3. $z_i^{\mathcal{R}} = 1$ if $\mathcal{R}$ stops at i; 0 otherwise. Analogously, $z_k^{\mathcal{S}} = 1$ if k is a stop of mode $\mathcal{S}$; 0 otherwise.

4. $f^w = 1$ if demand w uses $\mathcal{S}$, $\mathcal{R}$, and/or the combined modes RS and SR.

5. $f_a^{w\mathcal{R}} = 1$ if demand w traverses arc $a \in A_{\mathcal{R}}$, 0 otherwise.

6. $f_a^{w\mathcal{S}} = 1$ if demand w traverses arc $a \in A_{\mathcal{S}}$; 0 otherwise.

7. $f_k^{w\mathcal{SR}} = 1$ if demand w transfers from $\mathcal{S}$ to $\mathcal{R}$ at node $k \in N_{trans}$; 0 if there is no transfer of w from $\mathcal{S}$ to $\mathcal{R}$ at k.

8. $f_k^{w\mathcal{RS}} = 1$ if demand w transfers from $\mathcal{R}$ to $\mathcal{S}$ at node $k \in N_{trans}$; 0 if there is no transfer of w from $\mathcal{S}$ to $\mathcal{R}$ at k.

Objective function and constraints The aim of the problem is to design a rapid transit line $\mathcal{R}$ and to re-design the slow line $\mathcal{S}$ so that the trip coverage of both public modes would be maximized:

$$\max_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{f},\boldsymbol{f}^{\mathcal{R}},\boldsymbol{f}^{\mathcal{S}},\boldsymbol{f}^{\mathcal{RS}},\boldsymbol{f}^{\mathcal{SR}}} \sum_{w \in W} g^w f^w \tag{3.1}$$

Therefore, the private traffic is minimized.

- Budget constraints. An equivalent way to impose upper bounds on the budget is by upper bounding the number of edges that can form each line.

$$\sum_{e \in E_{\mathcal{R}}} x_e^{\mathcal{R}} \leq E_{\mathcal{R}}^{max}, \tag{3.2}$$

$$\sum_{e \in E_{\mathcal{S}}} x_e^{\mathcal{S}} \leq E_{\mathcal{S}}^{max}. \tag{3.3}$$

- Design constraints.

$$x_e^{\mathcal{R}} \leq z_i^{\mathcal{R}} + y_i^{\mathcal{R}}, \qquad e \in E_{\mathcal{R}}, \quad i \in e, \tag{3.4}$$

$$\sum_{o \in O_{\mathcal{R}}} \sum_{e \in \delta(o)} x_e^{\mathcal{R}} = 1, \tag{3.5}$$

$$\sum_{d \in D_{\mathcal{R}}} \sum_{e \in \delta(d)} x_e^{\mathcal{R}} = 1, \tag{3.6}$$

$$\sum_{o \in O_{\mathcal{R}}} z_o^{\mathcal{R}} = 1, \tag{3.7}$$

$$\sum_{d \in D_{\mathcal{R}}} z_d^{\mathcal{R}} = 1, \tag{3.8}$$

$$z_i^{\mathcal{R}} + y_i^{\mathcal{R}} \leq 1, \qquad i \in N_{\mathcal{R}}, \tag{3.9}$$

$$\sum_{e \in E_{\mathcal{R}}} x_e^{\mathcal{R}} + 1 = \sum_{i \in N_{\mathcal{R}}} (y_i^{\mathcal{R}} + z_i^{\mathcal{R}}), \tag{3.10}$$

$$\sum_{e\in\delta(k)} x_e^{\mathcal{R}} \leq 2(z_k^{\mathcal{R}} + y_k^{\mathcal{R}}), \qquad k \in N_{\mathcal{R}} \setminus (O_{\mathcal{R}} \cup D_{\mathcal{R}}), \quad (3.11)$$

$$\sum_{e\in\delta(i)} x_e^{\mathcal{R}} \leq |B| - 1, \qquad i \in B,\ B \subseteq N_{\mathcal{R}},\ |B| \geq 2, \quad (3.12)$$

$$x_e^{\mathcal{S}} \leq z_i^{\mathcal{S}}, \qquad e \in E_{\mathcal{S}}, \quad i \in e, \quad (3.13)$$

$$\sum_{o\in O_{\mathcal{S}}} z_o^{\mathcal{S}} = 1, \quad (3.14)$$

$$\sum_{d\in D_{\mathcal{S}}} z_d^{\mathcal{S}} = 1, \quad (3.15)$$

$$\sum_{e\in E_{\mathcal{S}}} v_e^{\mathcal{S}} x_e^{\mathcal{S}} \geq E_{\mathcal{S}}^{id}, \quad (3.16)$$

$$\sum_{e\in E_{\mathcal{S}}} x_e^{\mathcal{S}} + 1 = \sum_{i\in N_{\mathcal{S}}} z_i^{\mathcal{S}}, \quad (3.17)$$

$$\sum_{e\in\vartheta(k)} x_e^{\mathcal{S}} \leq 2z_k^{\mathcal{S}}, \qquad k \in N_{\mathcal{S}} \setminus (O_{\mathcal{S}} \cup D_{\mathcal{S}}), \quad (3.18)$$

$$\sum_{e\in\vartheta(i)} x_e^{\mathcal{S}} \leq |B| - 1, \qquad i \in B,\ B \subseteq N_{\mathcal{S}},\ |B| \geq 2, \quad (3.19)$$

$$f^w \leq 1 - y_{w^s}^{\mathcal{R}}, \qquad w = (w^s, w^t) \in W, \text{ if } w^s \in N_{\mathcal{R}} \text{ and } w^s \notin N_{\mathcal{S}}, \quad (3.20)$$

$$f^w \leq 1 - y_{w^t}^{\mathcal{R}}, \qquad w = (w^s, w^t) \in W, \text{ if } w^t \in N_{\mathcal{R}} \text{ and } w^s \notin N_{\mathcal{S}}, \quad (3.21)$$

$$f^w \leq \begin{cases} z_k^{\mathcal{R}} + z_k^{\mathcal{S}}, & \text{if } k \in N_{\mathcal{R}} \cap N_{\mathcal{S}}, \\ z_k^{\mathcal{R}}, & \text{if } k \in N_{\mathcal{R}} \text{ and } k \notin N_{\mathcal{S}}, \\ z_k^{\mathcal{S}}, & \text{if } k \in N_{\mathcal{S}} \text{ and } k \notin N_{\mathcal{R}}, \end{cases} \qquad w \in W, \quad k \in \{w^s, w^t\}. \quad (3.22)$$

Constraints (3.4) enforce the extremes of a constructed edge of line $\mathcal{R}$ to be stations (constructed nodes) or non-stop nodes. Constraints (3.5) and (3.6) guarantee that the end-nodes of $\mathcal{R}$ have only one incident edge. Constraints (3.7) and (3.8) impose that exactly one node has to be selected from the sets of possible origins and destinations of the rapid transit line (see Remark 10). Constraints (3.9) do not allow a node $i \in N_{\mathcal{R}}$ to be simultaneously a stop and non-stop node. Constraints (3.10)-(3.12) impose that the design of network $\mathcal{R}$ is a chain graph (see Remark 9). For the re-location of the slow line $\mathcal{S}$, we consider similar constraints. Constraints (3.13) enforce the extremes of a constructed edge of line $\mathcal{S}$ to be stop nodes. Constraints (3.14) and (3.15) impose that exactly one node has to be selected from the sets of possible origins and destinations of $\mathcal{S}$ (see Remark 10). Constraint (3.16) states that the old and new line $\mathcal{S}$ must coincide in a number of edges. Constraints (3.17)-(3.19) impose that the design of network $\mathcal{S}$ is a chain graph (see Remark 9). Constraints (3.20) and (3.21) impose that through a non-stop node in $\mathcal{R}$ only passes flow whose origin or destination is another node different from this one.

Constraints (3.22) enforce to the origin and destination of an O/D pair be station in $\mathcal{R}$ and/or stop node in $\mathcal{S}$ if its demand uses the public system.

- Flow conservation constraints.

$$\sum_{a\in\delta_w^+(k)} f_a^{w\mathcal{R}} + \sum_{a\in\vartheta_w^+(k)} f_a^{w\mathcal{S}} - \left(\sum_{a\in\delta_w^-(k)} f_a^{w\mathcal{R}} + \sum_{a\in\vartheta_w^-(k)} f_a^{w\mathcal{S}}\right) =$$
$$\begin{cases} f^w, \text{ if } k = w^s, \\ -f^w, \text{ if } k = w^t, \quad w \in W, \quad k \in N_{\mathcal{R}} \cup N_{\mathcal{S}} \text{ and } k \notin N_{trans} \setminus \{w^s, w^t\}. \\ 0, \text{ otherwise} \end{cases} \tag{3.23}$$

$$\sum_{a\in\delta_w^-(k)} f_a^{w\mathcal{R}} + f_k^{w\mathcal{S}\mathcal{R}} - \left(\sum_{a\in\delta_w^+(k)} f_a^{w\mathcal{R}} + f_k^{w\mathcal{R}\mathcal{S}}\right) = 0,$$
$$w \in W, \quad k \in N_{trans} \setminus \{w^s, w^t\}, \tag{3.24}$$

$$\sum_{a\in\vartheta_w^-(k)} f_a^{w\mathcal{S}} + f_k^{w\mathcal{R}\mathcal{S}} - \left(\sum_{a\in\vartheta_w^+(k)} f_a^{w\mathcal{S}} + f_k^{w\mathcal{S}\mathcal{R}}\right) = 0,$$
$$w \in W, \quad k \in N_{trans} \setminus \{w^s, w^t\}. \tag{3.25}$$

Constraints (3.23) are the usual flow conservation ones for both public systems. Besides, flow conservation constraints for transfer points (3.24) and (3.25) are needed.

- Location-allocation constraints. They do not allow flow on an edge of the lines $\mathcal{R}$ and $\mathcal{S}$ unless the edge has been built.

$$f_a^{w\mathcal{R}} + f_{\hat{a}}^{w\mathcal{R}} \le x_e^{\mathcal{R}}, \qquad w \in W, e = \{i,j\} \in E_{\mathcal{R}} : a = (i,j), \hat{a} = (j,i), \tag{3.26}$$

$$f_a^{w\mathcal{S}} + f_{\hat{a}}^{w\mathcal{S}} \le x_e^{\mathcal{S}}, \qquad w \in W, e = \{i,j\} \in E_{\mathcal{S}} : a = (i,j), \hat{a} = (j,i). \tag{3.27}$$

- Alignment stop constraints.

$$f_k^{w\mathcal{SR}} + f_k^{w\mathcal{RS}} \leq z_k^{\mathcal{R}}, \qquad w \in W, k \in N_{trans} \setminus \{w^s, w^t\}, \tag{3.28}$$

$$f_k^{w\mathcal{SR}} + f_k^{w\mathcal{RS}} \leq z_k^{\mathcal{S}}, \qquad w \in W, k \in N_{trans} \setminus \{w^s, w^t\}, \tag{3.29}$$

$$\sum_{a \in \delta^+(w^s)} f_a^{w\mathcal{R}} \leq z_{w^s}^{\mathcal{R}}, \qquad w = (w^s, w^t) \in W, \text{ if } w^s \in N_{\mathcal{R}}, \tag{3.30}$$

$$\sum_{a \in \delta^-(w^t)} f_a^{w\mathcal{R}} \leq z_{w^t}^{\mathcal{R}}, \qquad w = (w^s, w^t) \in W, \text{ if } w^t \in N_{\mathcal{R}}. \tag{3.31}$$

Constraints (3.28) and (3.29) locate stops whenever there is traffic demand entering or leaving one of the two modes. Constraints (3.30) and (3.31) state that a station in $\mathcal{R}$ has to be constructed in a node if the node has inflow or outflow, respectively.

- Mode choice. The demand is assigned either to the public modes or to the existing private one depending on the total cost (time or length) of the trip.

$$\begin{aligned}&\sum_{a \in A_{\mathcal{R}}} t_a^{\mathcal{R}} f_a^{w\mathcal{R}} + \sum_{a \in A_{\mathcal{S}}} t_a^{\mathcal{S}} f_a^{w\mathcal{S}} + \sum_{k \in N_{trans}} t_k^{\mathcal{RS}} f_k^{w\mathcal{RS}} + \sum_{k \in N_{trans}} t_k^{\mathcal{SR}} f_k^{w\mathcal{SR}} + \\ &+ t_{stop}^{\mathcal{R}} \sum_{k \in N_{\mathcal{R}}} z_k^{\mathcal{R}} \sum_{a \in \delta^+(k)} f_a^{w\mathcal{R}} + t_{stop}^{\mathcal{S}} \sum_{k \in N_{\mathcal{S}}} z_k^{\mathcal{S}} \sum_{a \in \vartheta^+(k)} f_a^{w\mathcal{S}} + \\ &+ f^w \left(t_{wait}^{\mathcal{R}} - \frac{1}{2} t_{stop}^{\mathcal{R}} \right) \leq u^w f^w, \qquad w \in W.\end{aligned} \tag{3.32}$$

- Binary constraints. All the variables are assumed to be in $\{0, 1\}$.

$$x_e^{\mathcal{R}},\, x_e^{\mathcal{S}},\, y_k^{\mathcal{R}},\, z_k^{\mathcal{R}},\, z_k^{\mathcal{S}},\, f^w,\, f_a^{w\mathcal{R}},\, f_a^{w\mathcal{S}},\, f_k^{w\mathcal{RS}},\, f_k^{w\mathcal{SR}} \in \{0, 1\}. \tag{3.33}$$

Remark 9. *Constraints* (3.12) *and* (3.19) *are the common ones for the sub-tours (cycles) elimination. These constraints have a high combinatorial burden. Then, in practice, they are not usually added to the formulation. Instead, when a solution contains cycles, additional constraints can be imposed to avoid the presence of those cycles in the solution network. However, it could be interesting to include the possibility of considering circular lines since well developed networks (such as Madrid or París) often contains one circular line at least.*

Remark 10. *For the case study, we have added to the formulation the constraints*

$$z_i^{\mathcal{R}} + z_i^{\mathcal{S}} \leq 1, \quad i \in (O_{\mathcal{R}} \cap O_{\mathcal{S}}) \cup (O_{\mathcal{R}} \cap D_{\mathcal{S}}) \cup (O_{\mathcal{S}} \cap D_{\mathcal{R}}) \cup (D_{\mathcal{R}} \cap D_{\mathcal{S}}), \qquad (3.34)$$

which ensure that the end-points of $\mathcal{R}$ and $\mathcal{S}$ must be different.

Limiting the number of transfers between the different lines To limit the number of transfers between lines $\mathcal{R}$ and $\mathcal{S}$, we have considered two options.

- On the one hand, using the set of variables $f_k^{w\mathcal{R}\mathcal{S}}$ and $f_k^{w\mathcal{S}\mathcal{R}}$. That is, for example, if we add to the formulation constraints

$$\sum_{k \in N_{trans} \setminus \{w^s, w^t\}} f_k^{w\mathcal{S}\mathcal{R}} \leq 1, \qquad w \in W, \qquad (3.35)$$

$$\sum_{k \in N_{trans} \setminus \{w^s, w^t\}} f_k^{w\mathcal{R}\mathcal{S}} \leq 1, \qquad w \in W, \qquad (3.36)$$

  we are limiting each commodity w to make at most two transfers, one from $\mathcal{R}$ to $\mathcal{S}$ and one from $\mathcal{S}$ to $\mathcal{R}$. If instead of these two previous costraints we add this one

$$\sum_{k \in N_{trans} \setminus \{w^s, w^t\}} f_k^{w\mathcal{R}\mathcal{S}} + f_k^{w\mathcal{S}\mathcal{R}} \leq 1, \qquad w \in W, \qquad (3.37)$$

  only one transfer is allowed for each commodity.

- On the other hand, limiting the number of transfers can also be done indirectly upper bounding the number of transfer stations to build. For that, we use the set of variables $z_i^{\mathcal{R}}$ and $z_i^{\mathcal{S}}$ where $i \in N_{trans}$. That is, we can add the following constraint

$$\sum_{i \in V} z_i^{\mathcal{R}} + z_i^{\mathcal{S}} \leq 2t + 1, \qquad V \subseteq N_{trans},\ |V| = t + 1, \qquad (3.38)$$

  where t refers to an upper bound for the number of transfer stations to build. Note that, if for a transfer station i, both variables $z_i^{\mathcal{R}}$ and $z_i^{\mathcal{S}}$ are set to 1, it means that i is an station which can be used by some commodity to tranfer to one line to the other.

For the case study detailed in Section 3.4, we have used the second option. It has been set to build at most one transfer station between $\mathcal{R}$ and **S**. Then, we have added to the formulation the set of constraints:

$$z_i^{\mathcal{R}} + z_i^{\mathcal{S}} + z_j^{\mathcal{R}} + z_j^{\mathcal{S}} \leq 3, \qquad i, j \in N_{trans},\ i \neq j. \qquad (3.39)$$

Linearization of terms $f_a^{\mathcal{R}} z_k^{\mathcal{R}}$ and $f_a^{\mathcal{S}} z_k^{\mathcal{S}}$ We define new variables $h_a^{\mathcal{R}}$ and $h_a^{\mathcal{S}}$ to replace the products of variables $f_a^{\mathcal{R}} z_k^{\mathcal{R}}$ and $f_a^{\mathcal{S}} z_k^{\mathcal{S}}$, respectively. Then, we add to the formulation the following sets of constraints.

$$\begin{aligned}
&h_a^{w\mathcal{R}} \leq f_a^{w\mathcal{R}}, && w \in W, a \in A_{\mathcal{R}},\\
&h_a^{w\mathcal{R}} \leq z_{a^s}^{\mathcal{R}}, && w \in W, a \in A_{\mathcal{R}},\\
&f_a^{w\mathcal{R}} - \left(1 - z_{a^s}^{\mathcal{R}}\right) \leq h_a^{w\mathcal{R}}, && w \in W, a \in A_{\mathcal{R}},\\
&h_a^{w\mathcal{R}} \in \{0,1\}, && w \in W, a \in A_{\mathcal{R}}.
\end{aligned} \tag{3.40}$$

$$\begin{aligned}
&h_a^{w\mathcal{S}} \leq f_a^{w\mathcal{S}}, && w \in W, a \in A_{\mathcal{S}},\\
&h_a^{w\mathcal{S}} \leq z_{a^s}^{\mathcal{S}}, && w \in W, a \in A_{\mathcal{S}},\\
&f_a^{w\mathcal{S}} - \left(1 - z_{a^s}^{\mathcal{S}}\right) \leq h_a^{w\mathcal{S}}, && w \in W, a \in A_{\mathcal{S}},\\
&h_a^{w\mathcal{S}} \in \{0,1\}, && w \in W, a \in A_{\mathcal{S}}.
\end{aligned} \tag{3.41}$$

Preprocessing methods It is always appropriate to apply preprocessing methods to reduce the data set in question. In this case, we have performed a very light preprocessing. For each w we have built a subgraph $\mathcal{N}^w = (N^w, E^w)$. Similar to the preprocessing in problems of Chapter 2, in each subgraph we only consider useful nodes and edges from $\mathcal{N}$. For the $(RTL\text{+}IFBS)$ problem, in each subgraph, such nodes and edges are those that belong to any path from w^s to w^t with a number of edges equal to or less than

- $E_{max}^{\mathcal{R}}$ if $w^s, w^t \in N^{\mathcal{R}}$ and $w^s, w^t \notin N^{\mathcal{S}}$,
- $E_{max}^{\mathcal{S}}$ if $w^s, w^t \in N^{\mathcal{S}}$ and $w^s, w^t \notin N^{\mathcal{R}}$,
- $E_{max}^{\mathcal{R}}$ if $w^s \in N^{\mathcal{R}} \cap N^{\mathcal{S}}, w^t \in N^{\mathcal{R}}$ and $w^t \notin N^{\mathcal{S}}$,
- $E_{max}^{\mathcal{S}}$ if $w^s \in N^{\mathcal{R}} \cap N^{\mathcal{S}}, w^t \in N^{\mathcal{S}}$ and $w^t \notin N^{\mathcal{R}}$,
- $E_{max}^{\mathcal{R}} + E_{max}^{\mathcal{S}} - 1$ if $w^s \in N^{\mathcal{R}}, w^s \notin N^{\mathcal{R}}, w^t \in N^{\mathcal{S}}$ and $w^t \notin N^{\mathcal{S}}$.

Note that the role of w^s and w^t is interchangeable.

3.3 Benders decomposition implementation

In this section we describe two Benders implementations to solve the problem. Actually, our Benders implementations are used as a subroutine in a Branch-and-Benders-cut scheme. By the Benders Theory, a set of continuous variables is needed. Proposition 4 shows that we can relax binary variables $f_a^{w\mathcal{R}}$, $f_a^{w\mathcal{S}}$, $f_k^{w\mathcal{RS}}$ and $f_k^{w\mathcal{SR}}$. Let

(RTL+$IFBS\_R$) denote the formulation (RTL+$IFBS$) in which the mentioned flow variables are replaced by non-negativity variables, i.e.

$$f_a^{w\mathcal{R}} \geq 0, \qquad w \in W, a \in A_{\mathcal{R}}, \tag{3.42}$$

$$f_a^{w\mathcal{S}} \geq 0, \qquad w \in W, a \in A_{\mathcal{S}}, \tag{3.43}$$

$$f_k^{w\mathcal{RS}},\, f_k^{w\mathcal{SR}} \geq 0, \qquad w \in W, a \in N_{trans}. \tag{3.44}$$

By Definition 1, the following proposition states.

Proposition 4. *The projections of* (RTL+$IFBS\_R$) *and* (RTL+$IFBS\_R$) *onto de* $(\boldsymbol{f^{\mathcal{R}}}, \boldsymbol{f^{\mathcal{S}}}, \boldsymbol{f^{\mathcal{RS}}}, \boldsymbol{f^{\mathcal{SR}}})$*-space coincide.*

$$Proj_{\boldsymbol{x^{\mathcal{R}}},\boldsymbol{x^{\mathcal{S}}},\boldsymbol{y^{\mathcal{R}}},\boldsymbol{z^{\mathcal{R}}},\boldsymbol{z^{\mathcal{S}}},\boldsymbol{f}}(\mathcal{F}(RTL\text{+}IFBS)) = Proj_{\boldsymbol{x^{\mathcal{R}}},\boldsymbol{x^{\mathcal{S}}},\boldsymbol{y^{\mathcal{R}}},\boldsymbol{z^{\mathcal{R}}},\boldsymbol{z^{\mathcal{S}}},\boldsymbol{f}}(\mathcal{F}(RTL\text{+}IFBS\_R)).$$

Proof. First, $\mathcal{F}(RTL\text{+}IFBS) \subseteq \mathcal{F}(RTL\text{+}IFBS\_R)$ implies $Proj_{\boldsymbol{x^{\mathcal{R}}},\boldsymbol{x^{\mathcal{S}}},\boldsymbol{y^{\mathcal{R}}},\boldsymbol{z^{\mathcal{R}}},\boldsymbol{z^{\mathcal{S}}},\boldsymbol{f}}(\mathcal{F}(RTL\text{+}IFBS)) \subseteq Proj_{\boldsymbol{x^{\mathcal{R}}},\boldsymbol{x^{\mathcal{S}}},\boldsymbol{y^{\mathcal{R}}},\boldsymbol{z^{\mathcal{R}}},\boldsymbol{z^{\mathcal{S}}},\boldsymbol{f}}(\mathcal{F}(RTL\text{+}IFBS\_R))$. Second, let $(\boldsymbol{x^{\mathcal{R}}}, \boldsymbol{x^{\mathcal{S}}}, \boldsymbol{y^{\mathcal{R}}}, \boldsymbol{z^{\mathcal{R}}}, \boldsymbol{z^{\mathcal{S}}}, \boldsymbol{f})$ be a point belonging to $Proj_{\boldsymbol{x^{\mathcal{R}}},\boldsymbol{x^{\mathcal{S}}},\boldsymbol{y^{\mathcal{R}}},\boldsymbol{z^{\mathcal{R}}},\boldsymbol{z^{\mathcal{S}}},\boldsymbol{f}}(\mathcal{F}(RTL\text{+}IFBS\_R))$. For every O/D pair $w \in W$ such that $f^w = 0$, then $\boldsymbol{f^{w\mathcal{R}}} = 0$, $\boldsymbol{f^{w\mathcal{S}}} = 0$, $\boldsymbol{f^{w\mathcal{RS}}} = 0$ and $\boldsymbol{f^{w\mathcal{SR}}} = 0$. In the case where $f^w = 1$, there exists a flow $\boldsymbol{f^{w\mathcal{R}}} > 0$, $\boldsymbol{f^{w\mathcal{S}}} > 0$, $\boldsymbol{f^{w\mathcal{RS}}} > 0$ and $\boldsymbol{f^{w\mathcal{SR}}} > 0$ satisfying (3.23)-(3.32) that can be decomposed into a convex combination of flows on paths from w^s to w^t and cycles. Due to the fact that the end-points of $\mathcal{R}$ are different from those of $\mathcal{S}$ and that there exists one tranfer station at most, there is only one feasible path for such flow. Hence, by taking $f_a^{w\mathcal{R}}$, $f_a^{w\mathcal{S}}$, $f_k^{w\mathcal{RS}}$ and $f_a^{w\mathcal{SR}}$ equal to 1 for the arcs and transfer stations belonging to this path and to 0 otherwise, we show that $(\boldsymbol{x^{\mathcal{R}}}, \boldsymbol{x^{\mathcal{S}}}, \boldsymbol{y^{\mathcal{R}}}, \boldsymbol{z^{\mathcal{R}}}, \boldsymbol{z^{\mathcal{S}}}, \boldsymbol{f})$ also belongs to $Proj_{\boldsymbol{x^{\mathcal{R}}},\boldsymbol{x^{\mathcal{S}}},\boldsymbol{y^{\mathcal{R}}},\boldsymbol{z^{\mathcal{R}}},\boldsymbol{z^{\mathcal{S}}},\boldsymbol{f}}(\mathcal{F}(RTL\text{+}IFBS))$. □

The master problem that we solve is:

$$\begin{aligned}
(M\_RTL\text{+}IFBS) \quad & \max_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z},\boldsymbol{f}} \sum_{w \in W} g^w f^w \\
& \text{s.t. } (3.2)\text{-}(3.22) \\
& \quad + \{\text{Benders Cuts } (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{f})\} \\
& \quad x_e^{\mathcal{R}},\, x_e^{\mathcal{S}},\, y_k^{\mathcal{R}},\, z_k^{\mathcal{R}},\, z_k^{\mathcal{S}},\, f^w, h_a^{w\mathcal{R}}, h_a^{w\mathcal{S}} \in \{0, 1\}.
\end{aligned} \tag{3.45}$$

In Section 3.3.1, we show the standard Benders cuts obtained by dualizing the respective feasibility subproblem. Then, in Section 3.3.2, we use the ideas exposed in Conforti and Wolsey (2019) in order to get stronger cuts than the standard ones. Both approaches have been fully detailed for two covering problems in Chapter 2. The situations that arise when applying Benders Theory are quite similar to the other problems studied. For this reason, in the next two subsections we will only show the formulations and the cuts derived from Benders Theory. For more details see Subsection 2.3.3.

3.3.1 LP feasibility cuts

Since the structure of the model allows it, we consider a feasibility subproblem made of constraints (3.23)-(3.32), (3.40) and (3.41) for each commodity $w \in W$, denoted by $(SP)^w$. As it is clear from the context, we remove the index w from the notation. Then, the dual of each feasibility subproblem can be expressed as follows.

Firstly, the objective function is:

$$\begin{aligned}
\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta},\boldsymbol{\sigma},\boldsymbol{\eta},\boldsymbol{\nu},\boldsymbol{\theta}} \; & f\,\alpha_{w^s} - \sum_{e\in E_{\mathcal{R}}} x_e^{\mathcal{R}}\,\sigma_e^{\mathcal{R}} - \sum_{e\in E_{\mathcal{S}}} x_e^{\mathcal{S}}\,\sigma_e^{\mathcal{S}} - \nu^{\mathcal{SR}} - \nu^{\mathcal{RS}} - \\
& - \sum_{a\in A_{\mathcal{R}}} f\,\delta_a^{\mathcal{R}} - \sum_{a\in A_{\mathcal{S}}} f\,\delta_a^{\mathcal{S}} - \sum_{k\in N_{trans}\setminus\{w^s,w^t\}} z_k^{\mathcal{R}}\,\theta_k^{\mathcal{R}} - \sum_{k\in N_{trans}\setminus\{w^s,w^t\}} z_k^{\mathcal{S}}\,\theta_k^{\mathcal{S}} - \\
& - \left(u - f\left(t_{wait}^{\mathcal{R}} - \frac{1}{2}\,t_{stop}^{\mathcal{R}}\right) - t_{stop}^{\mathcal{R}} \sum_{a\in A_{\mathcal{R}}} h_a^{\mathcal{R}} - t_{stop}^{\mathcal{S}} \sum_{a\in A_{\mathcal{S}}} h_a^{\mathcal{S}} \right)\eta + \\
& + \sum_{a\in A_{\mathcal{R}}} h_a^{\mathcal{R}}\,\lambda_a^{\mathcal{R}} + \sum_{a\in A_{\mathcal{S}}} h_a^{\mathcal{S}}\,\lambda_a^{\mathcal{S}} + \sum_{a\in A_{\mathcal{R}}} \left(h_a^{\mathcal{R}} - z_{a^s}^{\mathcal{R}}\right)\mu_a^{\mathcal{R}} + \sum_{a\in A_{\mathcal{S}}} \left(h_a^{\mathcal{S}} - z_{a^s}^{\mathcal{S}}\right)\mu_a^{\mathcal{S}} - \\
& - \sum_{a\in A_{\mathcal{R}}} \left(1 + h_a^{\mathcal{R}} - z_{a^s}^{\mathcal{R}}\right)\rho_a^{\mathcal{R}} - \sum_{a\in A_{\mathcal{S}}} \left(1 + h_a^{\mathcal{S}} - z_{a^s}^{\mathcal{S}}\right)\rho_a^{\mathcal{S}}
\end{aligned} \tag{3.46}$$

Besides, if $w^s \in N_{\mathcal{R}}$, the term $-\,z_{w^s}^{\mathcal{R}}\,\gamma_{w^s}^{\mathcal{R}}$ is added to the objective function. In the same way, if $w^t \in N_{\mathcal{R}}$, we add $-\,z_{w^t}^{\mathcal{R}}\,\gamma_{w^t}^{\mathcal{R}}$.

With respect to the constraints:

$$\left.\begin{array}{l}
\alpha_i - \alpha_j - \sigma_e^{\mathcal{R}} - \delta_a^{\mathcal{R}} - t_a^{\mathcal{R}}\,\eta + \lambda_a^{\mathcal{R}} - \rho_a^{\mathcal{R}} \\
\circ \text{ if } j \in N_{trans}\setminus\{w^s,w^t\}: \quad +\beta_j^{\mathcal{R}} \\
\circ \text{ if } i \in N_{trans}\setminus\{w^s,w^t\}: \quad -\beta_i^{\mathcal{R}} \\
\circ \text{ if } i = w^s: \quad -\gamma_{w^s}^{\mathcal{R}} \\
\circ \text{ if } j = w^t: \quad -\gamma_{w^t}^{\mathcal{R}}
\end{array}\right\} \leq 0, \quad a=(i,j)\in A_{\mathcal{R}} : e=\{i,j\}, \tag{3.47}$$

$$\left.\begin{array}{l}
\alpha_i - \alpha_j - \sigma_e^{\mathcal{S}} - \delta_a^{\mathcal{S}} - t_a^{\mathcal{S}}\,\eta + \lambda_a^{\mathcal{S}} - \rho_a^{\mathcal{S}} \\
\circ \text{ if } j \in N_{trans}\setminus\{w^s,w^t\}: \quad +\beta_j^{\mathcal{S}} \\
\circ \text{ if } i \in N_{trans}\setminus\{w^s,w^t\}: \quad -\beta_i^{\mathcal{S}}
\end{array}\right\} \leq 0, \quad a=(i,j)\in A_{\mathcal{S}} : e=\{i,j\}, \tag{3.48}$$

$$-\,\nu^{\mathcal{SR}} + \beta_k^{\mathcal{R}} - \beta_k^{\mathcal{S}} - \theta_k^{\mathcal{R}} - \theta_k^{\mathcal{S}} - t_k^{\mathcal{SR}}\,\eta \leq 0, \quad k \in N_{trans}\setminus\{w^s,w^t\}, \tag{3.49}$$

$$-\,\nu^{\mathcal{RS}} + \beta_k^{\mathcal{S}} - \beta_k^{\mathcal{R}} - \theta_k^{\mathcal{R}} - \theta_k^{\mathcal{S}} - t_k^{\mathcal{RS}}\,\eta \leq 0, \quad k \in N_{trans}\setminus\{w^s,w^t\}, \tag{3.50}$$

$$\gamma_i^{\mathcal{R}},\ \sigma_e^{\mathcal{R}},\ \sigma_e^{\mathcal{S}},\ \delta_e^{\mathcal{R}},\ \delta_e^{\mathcal{S}},\ \theta_i^{\mathcal{R}},\ \theta_i^{\mathcal{S}},\ \nu^{\mathcal{RS}},\ \nu^{\mathcal{SR}},\ \eta \geq 0. \tag{3.51}$$

Hence, due to the feasibility structure of $(SP)^w$, only feasibility Benders cuts will be needed:

$$\left.\begin{array}{l}
\left(\alpha_{w^s} - \displaystyle\sum_{a\in A_{\mathcal{R}}} \delta_a^{\mathcal{R}} - \sum_{a\in A_{\mathcal{S}}} \delta_a^{\mathcal{S}} + \left(t_{wait}^{\mathcal{R}} - \frac{1}{2}\, t_{stop}^{\mathcal{R}}\right)\eta\right) f \\
- \displaystyle\sum_{e\in E_{\mathcal{R}}} \sigma_e^{\mathcal{R}}\, x_e^{\mathcal{R}} - \sum_{e\in E_{\mathcal{S}}} \sigma_e^{\mathcal{S}}\, x_e^{\mathcal{S}} \\
+ \displaystyle\sum_{a\in A_{\mathcal{R}}} \left(\lambda_a^{\mathcal{R}} + t_{stop}^{\mathcal{R}} + \mu_a^{\mathcal{R}} - \rho_a^{\mathcal{R}}\right) h_a^{\mathcal{R}} \\
+ \displaystyle\sum_{a\in A_{\mathcal{S}}} \left(\lambda_a^{\mathcal{S}} + t_{stop}^{\mathcal{S}} + \mu_a^{\mathcal{S}} - \rho_a^{\mathcal{S}}\right) h_a^{\mathcal{S}} \\
+ \displaystyle\sum_{k\in N_{trans}\setminus\{w^s,w^t\}} \left(-\theta_k^{\mathcal{R}} + \sum_{a\in\delta^+(k)} (\rho_a^{\mathcal{R}} - \mu_a^{\mathcal{R}})\right) z_k^{\mathcal{R}} \\
+ \displaystyle\sum_{k\in N_{\mathcal{R}}\setminus\{N_{trans}\setminus\{w^s,w^t\}\}} \left(\sum_{a\in\delta^+(k)} (\rho_a^{\mathcal{R}} - \mu_a^{\mathcal{R}})\right) z_k^{\mathcal{R}} \\
+ \displaystyle\sum_{k\in N_{trans}\setminus\{w^s,w^t\}} \left(-\theta_k^{\mathcal{S}} + \sum_{a\in\gamma^+(k)} (\rho_a^{\mathcal{S}} - \mu_a^{\mathcal{S}})\right) z_k^{\mathcal{S}} \\
+ \displaystyle\sum_{k\in N_{\mathcal{S}}\setminus\{N_{trans}\setminus\{w^s,w^t\}\}} \left(\sum_{a\in\gamma^+(k)} (\rho_a^{\mathcal{S}} - \mu_a^{\mathcal{S}})\right) z_k^{\mathcal{S}} \\
\\
\circ\ \text{if } w^s \in N_{\mathcal{R}}: \quad -\gamma_{w^s}^{\mathcal{R}}\, z_{w^s}^{\mathcal{R}} \\
\circ\ \text{if } w^t \in N_{\mathcal{R}}: \quad -\gamma_{w^t}^{\mathcal{R}}\, z_{w^t}^{\mathcal{R}}
\end{array}\right\} \leq C, \tag{3.52}$$

where C has the expression $u\,\eta + \nu^{\mathcal{SR}} + \nu^{\mathcal{RS}} + \sum_{a\in A_{\mathcal{R}}} \rho_a^{\mathcal{R}} + \sum_{a\in A_{\mathcal{S}}} \rho_a^{\mathcal{S}}$.

3.3.2 Facet-defining Benders cuts

By the theory detailed in Subsection 2.3.3, an interior point $(x^{\mathcal{R},in}, x^{\mathcal{S},in}, y^{\mathcal{R},in}, z^{\mathcal{R},in} z^{\mathcal{S},in}, f^{in})$ of the convex hull of feasible solutions and a solution of the LP relaxation of the current restricted master problem $(x^{\mathcal{R},out}, x^{\mathcal{S},out}, y^{\mathcal{R},out}, z^{\mathcal{R},out} z^{\mathcal{S},out}, f^{out})$ are needed. In this situation, the expression of the dual subproblem considered depends on them.

Firstly, the objective function is:

$$\begin{aligned}
&\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\boldsymbol{\delta},\boldsymbol{\sigma},\boldsymbol{\eta},\boldsymbol{\nu},\boldsymbol{\theta}} f^{out}\,\alpha_{w^s} - \sum_{e\in E_{\mathcal{R}}} x_e^{\mathcal{R},out}\,\sigma_e^{\mathcal{R}} - \sum_{e\in E_{\mathcal{S}}} x_e^{\mathcal{S},out}\,\sigma_e^{\mathcal{S}} - \nu^{\mathcal{SR}} - \nu^{\mathcal{RS}} - \\
&- \sum_{a\in A_{\mathcal{R}}} f^{out}\,\delta_a^{\mathcal{R}} - \sum_{a\in A_{\mathcal{S}}} f^{out}\,\delta_a^{\mathcal{S}} - \sum_{k\in N_{trans}\setminus\{w^s,w^t\}} z_k^{R,out}\,\theta_k^{\mathcal{R}} - \\
&- \sum_{k\in N_{trans}\setminus\{w^s,w^t\}} z_k^{\mathcal{S},out}\,\theta_k^{\mathcal{S}} - \\
&- \left(u - f^{out}\left(t_{wait}^{\mathcal{R}} - \frac{1}{2}\,t_{stop}^{\mathcal{R}}\right) - t_{stop}^{\mathcal{R}} \sum_{a\in A_{\mathcal{R}}} h_a^{\mathcal{R},out} - t_{stop}^{\mathcal{S}} \sum_{a\in A_{\mathcal{S}}} h_a^{\mathcal{S},out}\right)\eta + \qquad (3.53)\\
&+ \sum_{a\in A_{\mathcal{R}}} h_a^{\mathcal{R},out}\,\lambda_a^{\mathcal{R}} + \sum_{a\in A_{\mathcal{S}}} h_a^{\mathcal{S},out}\,\lambda_a^{\mathcal{S}} + \sum_{a\in A_{\mathcal{R}}} \left(h_a^{\mathcal{R},out} - z_{a^s}^{\mathcal{R},out}\right)\mu_a^{\mathcal{R}} + \\
&+ \sum_{a\in A_{\mathcal{S}}} \left(h_a^{\mathcal{S},out} - z_{a^s}^{\mathcal{S},out}\right)\mu_a^{\mathcal{S}} - \sum_{a\in A_{\mathcal{R}}} \left(1 + h_a^{\mathcal{R},out} - z_{a^s}^{\mathcal{R},out}\right)\rho_a^{\mathcal{R}} - \\
&- \sum_{a\in A_{\mathcal{S}}} \left(1 + h_a^{\mathcal{S},out} - z_{a^s}^{\mathcal{S},out}\right)\rho_a^{\mathcal{S}}
\end{aligned}$$

Besides, if $w^s \in N_{\mathcal{R}}$, the term $-\,z_{w^s}^{\mathcal{R}}\,\gamma_{w^s}^{\mathcal{R}}$ is added to the objective function. In the same way, if $w^t \in N_{\mathcal{R}}$, we add $-\,z_{w^t}^{\mathcal{R}}\,\gamma_{w^t}^{\mathcal{R}}$.

With respect to the set of constraints, we consider constraints (3.47)-(3.51) together with the following normalization constraint:

$$\left.\begin{aligned}
&\Delta\, f\alpha_{w^s} - \sum_{e\in E_{\mathcal{R}}} \Delta\, x_e^{\mathcal{R}}\sigma_e^{\mathcal{R}} - \sum_{e\in E_{\mathcal{S}}} \Delta\, x_e^{\mathcal{S}}\sigma_e^{\mathcal{S}} - \sum_{a\in A_{\mathcal{R}}} \Delta\, f\,\delta_a^{\mathcal{R}} - \sum_{a\in A_{\mathcal{S}}} \Delta\, f\delta_a^{\mathcal{R}} \\
&- \sum_{k\in N_{trans}\setminus\{w^s,w^t\}} \Delta\, z_k^{\mathcal{R}}\theta_k^{\mathcal{R}} - \sum_{k\in N_{trans}\setminus\{w^s,w^t\}} \Delta\, z_k^{\mathcal{S}}\theta_k^{\mathcal{S}} \\
&+ \left(\left(t_{wait}^{\mathcal{R}} - \frac{1}{2}t_{stop}^{\mathcal{R}}\right)\Delta\, f + t_{stop}^{\mathcal{R}} \sum_{k\in N_{\mathcal{R}}}\sum_{a\in\delta^+(k)} \Delta\, h_a^{\mathcal{R}} + t_{stop}^{\mathcal{S}} \sum_{k\in N_{\mathcal{S}}}\sum_{a\in\gamma^+(k)} \Delta\, h_a^{\mathcal{S}}\right)\eta \\
&+ \sum_{a\in A_{\mathcal{R}}} \Delta h_a^{\mathcal{R}}\lambda_a^{\mathcal{R}} + \sum_{a\in A_{\mathcal{S}}} \Delta h_a^{\mathcal{S}}\lambda_a^{\mathcal{S}} + \sum_{a\in A_{\mathcal{R}}} \left(\Delta h_a^{\mathcal{R}} - \Delta z_{a^s}^{\mathcal{R}}\right)\mu_a^{\mathcal{R}} \\
&+ \sum_{a\in A_{\mathcal{S}}} \left(\Delta h_a^{\mathcal{S}} - \Delta z_{a^s}^{\mathcal{S}}\right)\mu_a^{\mathcal{S}} + \sum_{a\in A_{\mathcal{R}}} \left(\Delta z_{a^s}^{\mathcal{R}} - \Delta h_a^{\mathcal{R}}\right)\rho_a^{\mathcal{R}} + \sum_{a\in A_{\mathcal{S}}} \left(\Delta z_{a^s}^{\mathcal{S}} - \Delta h_a^{\mathcal{S}}\right)\rho_a^{\mathcal{S}} \\
&\circ\ \text{if } w^s \in N_{\mathcal{R}}: \quad -\Delta z_{w^s}^{\mathcal{R}}\gamma_{w^s}^{\mathcal{R}} \\
&\circ\ \text{if } w^t \in N_{\mathcal{R}}: \quad -\Delta z_{w^t}^{\mathcal{R}}\gamma_{w^t}^{\mathcal{R}}
\end{aligned}\right\} \leq 1$$

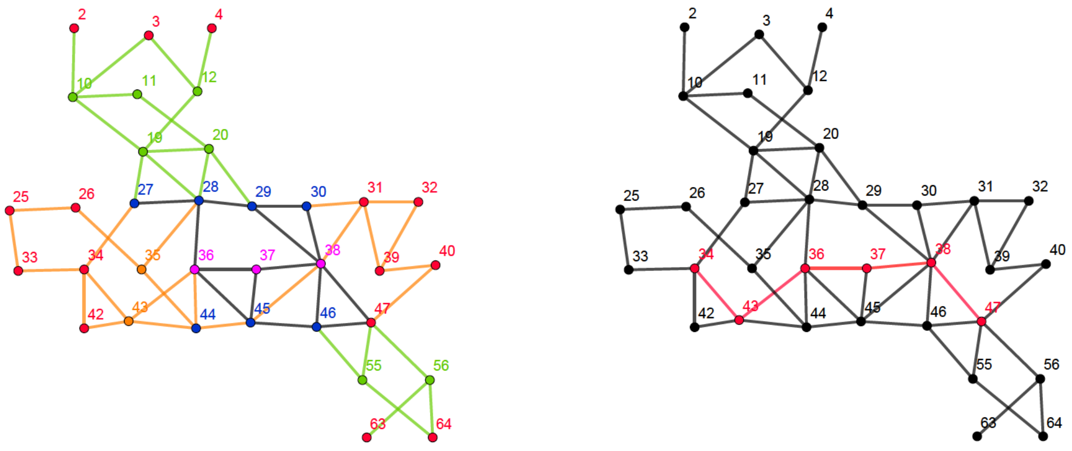
(3.54)

3.4 Computational experiments

Our computational experiments were performed on a computer equipped with a Intel Core i5-7300 CPU processor, with 2.50 gigahertz 4-core, and 16 gigabytes of RAM. The operating system is 64-bit Windows 10. Codes were implemented in Python 3.8. These experiments have been carried out through `CPLEX 12.10` solver, using its Python interface. `CPLEX` parameters were set to their default values and the model was optimized in a single threaded mode.

The tested instance is similar to the one used in González-Blanco et al. (2021). It is considered a subgraph of the `Seville` city network (composed of 64 nodes and 128 edges), shown in Figure 3.1a. Green nodes and edges form the sets $N_{\mathcal{R}}$ and $E_{\mathcal{R}}$. Orange nodes and edges form the sets $N_{\mathcal{S}}$ and $E_{\mathcal{S}}$. Nodes in red represent the set of possible starting and end stations of both modes. Blue nodes and black edges are those that belong to both modes of transport. Pink nodes represent the transfer stations. Line graph in red of Figure 3.1b represents the old slow line.

With respect to the parameters, the new slow line $\mathcal{S}$ must coincide with the old one on at least 3 edges, $E^{id}_{\mathcal{S}} = 3$, and can consist of a maximum of 6 edges in total. With respect to the rapid transit line, it must be composed of 10 nodes and 9 edges at most. The W set is formed by all possible O/D pairs. For the rest of the parameters see the suplementary material in `https://github.com/Natividad13/Network_Design_Benchmark_Instance`.



(a) Underlying network (b) Red line denotes the old slow line

Figure 3.1: Instance tested.

To apply the facet-defining Benders cuts approach we need an interior point. In this case, we have got it by solving the relaxed version of $(RTL{+}IFBS)$ using `CPLEX` and setting its parameter `barrier.crossover` to -1.

Regarding the computational time, after two hours, the optimal solution is obtained using the implemented Branch-and-Benders-cut scheme ad-hoc to the problem which generates facet-defining Benders cuts (named as `BD_CW`). That is, an hour and a half earlier than if we directly solve the MIP formulation with `CPLEX`. It should be noted that the implementation which generates the standard Benders cuts (`BD_Trd`) and the Benders decomposition algorithm existing in `CPLEX` (`Auto_BD`) are not competitive in this case (see Table 3.1).

`CPLEX`	`Auto_BD`		`BD_Trd`		`BD_CW`	
`t`	`t`	`cuts`	`t`	`cuts`	`t`	`cuts`
8244.44	-		36482.63	2863	5743.59	1103

Table 3.1: Computing times to compare the performance of Benders cuts for (RTL+$IFBS$).

This integrated model results in an optimum design with respect to the maximization of the coverage for the whole public transport system (composed by the rapid and slow modes). Locating each line independently, without taking into account the influence that may exist between them, or even locating them in a sequential way, can result in suboptimal solutions. The sequential design method is the one used in practice. That is, currently the rapid transit line $\mathcal{R}$ is located first and then the slow line $\mathcal{S}$ is re-located. For example, considering the tested instance, the optimal objective value for the integrated model is 62.02% and 72.17% bigger than those of the sequential and independently location models, respectively. Figure 3.2 shows the optimal solutions for the integrated and sequential models respectively.

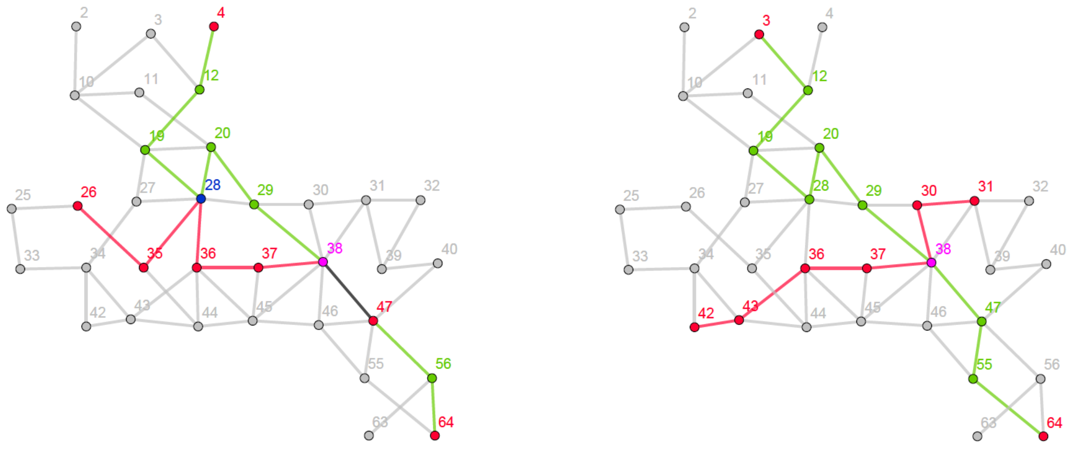


Figure 3.2: Optimal solutions for the sequential and integrated models to maximize the coverage.

3.5 Conclusions

In this chapter we have presented a formulation which locates a rapid transit line and modifies the old route of a slow transit line simultaneously with the purpose of maximizing the demand covered. We have focused on the case study in which a single transfer station is allowed and the end-points of the lines do not coincide. In addition, it is shown in an example with the `Seville` network instance that the design of both lines in an integrated way obtains greater coverage than that of sequentally designing (first the rapid transit line is designed and then the slow line is re-located).

To solve this problem, we have developed a branch-and-Benders-cut scheme specifically for the formulation discussed. Using the `Seville` network instance, we have obtained that the Benders approach that uses the ideas of Conforti and Wolsey (2019) is competitive against the standard decomposition of Benders (Subsection 3.3.1), the Benders procedure existing in `CPLEX` and that of directly using `CPLEX`.

Further research will go in the direction of considering the design of more than one rapid transit line as well as the re-location of more than one slow transit line in an integrated way. It will be interesting to explore the solution depending on the presence or not of cycles between the set of lines and the maximum number of transfer stations to be considered. Furthermore, developing a Benders decomposition to generate facet-defining Benders cuts for this much more complex case could be also efficient and advantageous.

Chapter 4

Metaheuristic approaches for the Maximal-Covering Network Design Problem

In this chapter we develop and test two metaheuristic techniques for the Maximal-Covering Network Design problem presented in Chapter 2. One of them is a Simulated Annealing algorithm and the other is an Adaptive Large Neighborhood Search procedure. Besides, we also test the Genetic Algorithm exposed in Perea et al. (2020). These three routines start from an already known initial solution. To compute this solution we have used the GRASP described in Perea et al. (2020). An extensive computational study is dedicated to each one of them using medium size randomly generated instances. Eventually, we compute these metaheuristics for a large instance.

4.1 Introduction

The current concept of metaheuristic techniques is based on the different interpretations of what is an intelligent way of solving a problem. Metaheuristics are clever strategies to design or improve very general heuristic procedures with high performance. Several authors have contributed in books and articles with interleaved reflections on a rigorous definition of the concept of metaheuristic, a structure and exhaustive classification of the different strategies and on the complete study of the characteristics of each one (see, for example, Lee and El-Sharkawi (2008), Potvin and Gendreau (2019)).

Metaheuristics for Network Design problems

Since most Network Design problems are NP-hard, some research efforts have been oriented to apply metaheuristic algorithms to obtain good though not necessarily optimal solutions, within a reasonable computational time. For instance, in the fields of Transportation and Telecommunications Network Design, Tabu Search (Pedersen et al. (2009), Xu et al. (1996)), Genetic Algorithms (Król and Król (2019), Perea et al. (2020), Chou et al. (2001)), Simulated Annealing (Fan and Machemehl (2006), Kermansshahi et al. (2010), Girgis et al. (2014)), Greedy Randomized Adaptive Search Procedures (García-Archilla et al. (2013), Maya Duque and Sörensen (2011), Risso and Robledo (2013)) and Adaptive Large Neighborhood Search algorithms (Canca et al. (2016), Canca et al. (2017), Zhang et al. (2022), Mehta et al. (2015)) have been applied to medium-size instances. Besides, matheuristics procedures have been also exploited (see Canca et al. (2019), Chouman and Crainic (2015), Wu et al. (2020)).

Constructive and improvement algorithms

As presented in Ahuja et al. (2002), the literature devoted to heuristic algorithms often distinguishes between two broad classes: *constructive algorithms* and *improvement algorithms.* A constructive algorithm builds a solution from scratch by assigning values to one or more decision variables at each iteration, just like the Greedy Randomized Adaptive Search Procedures do. An improvement algorithm generally starts with an initial (feasible or not) solution and tries to obtain a better solution in a iterative manner. Genetics and Neighborhood Search (also named Local Search) algorithms are wide classes of improvement algorithms.

Structure of this chapter

In Section 4.2 we describe the Greedy Randomized Adaptive Search Procedure from García-Archilla et al. (2013), used in this chapter to compute initial feasible solutions that will be improved. Then, in Sections 4.3 and 4.4 Simulated Annealing and Adaptive Large Neigborhood Search schemes are designed for the (MC) problem. In Section 4.5, the Genetic algorithm developed in Perea et al. (2020) for this problem is also presented. Further, in Subsection 4.6.2, a sensitivity analysis for the construction algorithm of Section 4.2 is done. Then, using the best configuration identified for it, in Subsections 4.6.3, 4.6.4 and 4.6.5, the other three improvement metaheuristics are analyzed in detail in order to evaluate their performance based on the parameters that make them up to improve solutions. Finally, in Subsection 4.6.6 they are compared using a large instance.

4.2 Determination of an initial solution with a GRASP

The metaheuristics presented in the next sections of this chapter start from a given initial feasible solution. The idea is to get this solution from a construction algorithm. In this case, for the (MC) problem, a *Greedy Randomized Adaptive Search Procedure (GRASP)* has been chosen; specifically, the one presented in García-Archilla et al. (2013).

A GRASP is a *multi-start* metaheuristic. That is, each iteration consists of two phases. The first one, also named as *construction phase*, generates a solution from scratch. The second one consists of a local search procedure. This phase is refered also to as *improvement phase* and it typically (but not necessarily) improves the solution found above. In other words, the construction phase builds a feasible solution whose neighborhood is investigated until a local optimum is found during the local search phase.

Regarding the GRASP developed in García-Archilla et al. (2013), its construction phase consists of adding one edge per iteration to an empty initial graph satisfying the budget constraint until no more edges can be added. At each iteration, the set of feasible edges is updated. An edge is said to be a *feasible edge* if it has not been built previously and its cost (plus that of its end-nodes if they are not built) is equal to or lower than the remaining available budget. Next, this set is ordered according to the objective value when each edge is individually added to the network under construction. From this set, the $k \geq 2$ edges that contribute the greatest are selected. Then, one of them is randomly chosen to be added to the network under construction. This procedure is repeated until the set of feasible edges is empty for some iteration reached. This routine is described in Algorithm 4. Further, the improvement phase works as follows. For each edge e in the current solution network, a new network is built by replacing e with a set of feasible non-built edges. These edges are added using the same criterion of the construction phase. The iteration finishes by picking the best solution obtained and comparing it with the solution obtained in the construction phase. This procedure is depicted in Algorithm 5. This randomized Local Search scheme is applied n_{iter_max} times.

With respect to the neighborhood, note that at each iteration no connected graphs could be generated. In García-Archilla et al. (2013), three possibilities about the neighborhood were taken into account: generate only connected graphs, generate graphs that might not be connected, or random generation.

In Subsection 4.6.2, a sensitivity analysis of the parameters of this GRASP is tested using random instances.

Algorithm 4: Constructive phase of GRASP

```
Require: $\bar{\mathcal{N}} = (\bar{N}, \bar{E})$ with $\bar{N} = \emptyset$ and $\bar{E} = \emptyset$, $\bar{W} = \emptyset$, $C = 0$.
Compute $FE = \{e = \{i, j\} \in E \setminus \bar{E} \,|\, c_e + b_i + b_j{}^1 \leq \alpha TC - C\}$.
while $FE \neq \emptyset$ do
    if $|FE| > k$ then
        Determine a set $SE = \{e_1, \dots, e_k\}$ consisting of the $k$ edges on $FE$
          that generate the largest objective value when they are individually
          added to $\bar{E}$.
    else
        Set $SE = FE$
    end
    Randomly choose one edge $e^* = \{i^*, j^*\} \in SE$ to be added to $\bar{\mathcal{N}}$.
    Update $\bar{\mathcal{N}}$, $FE$, $C$.
end
for $w \in W$ do
    Compute the shortest path for $w$ in $\bar{\mathcal{N}} = (\bar{N}, \bar{E})$.
    if its length is equal or shorter than $u^w$ then
        $\bar{W} = \bar{W} \cup \{w\}$
    end
end
return $\bar{\mathcal{N}}$, $\bar{W}$.
```

Algorithm 5: Improvement phase of GRASP

```
Require: $\bar{\mathcal{N}} = (\bar{N}, \bar{E})$, $\bar{W} = \emptyset$, $C_{imp} = C$.
Compute $\bar{E}^C = E \setminus \bar{E}$.
for $\bar{e} \in \bar{E}$ do
    Define $\bar{\mathcal{N}}_{\bar{e}} = (\bar{N}^1, \bar{E} \setminus \{\bar{e}\})$
    $B_{\bar{e}} = \alpha TC - C_{imp} - c_{\bar{e}} - c_{\bar{i}} - c_{\bar{j}}$
    Compute $FE_{\bar{e}} = \{e = \{i, j\} \in \bar{E}^C \setminus \bar{E} \,|\, c_e + b_i + {b_j}^2 \leq B_{\bar{e}}\}$
    while $FE_{\bar{e}} \neq \emptyset$ do
        Determine the edge $\tilde{e} = \{\tilde{i}, \tilde{j}\} \in FE_{\bar{e}}$ that gives the largest
          improvement in the objective function when added to $\mathcal{N}_{\bar{e}}$.
        Update $\bar{\mathcal{N}}_{\bar{e}}$, $FE_{\bar{e}}, B_{\bar{e}}$.
    end
end
Set $\bar{\mathcal{N}}^*$ the network that gives the largest improvement in the objective
  function among all $\bar{\mathcal{N}}_{\bar{e}}$, $\bar{e} \in \bar{E}$.
return $\bar{\mathcal{N}}^*$, $\bar{W}$.
```

4.3 Simulated Annealing

Annealing is the process of subjecting a solid to a high temperature with a subsequent cooling until it converges to a low energy state (that is, a steady frozen state). An example would be to obtain high-quality crystals from the molten solid. In the early 1950s, N. Metropolis proposed an algorithm that simulates this process of annealing. Later, in the 1980s, two independent researches were done by Kirkpatrick, Gelatt and Vecchi (Kirkpatrick et al. (1983)) and Čerňy (Čerňy (1985)). They adapted the Metropolis algorithm in order to be useful for solving combinatorial problems due to the fact that they noted the similarities between the physical process and these problems (Monticelli et al. (2008)).

As explained in Eglese (1990), the Metropolis algorithm simulates the behaviour of a set of atoms while the temperature decreases. At each iteration, a random atom is displaced and it causes a change in the energy of the system, δ. If $\delta < 0$, the resulting change is accepted, but if $\delta > 0$, this change is accepted with probability $\exp(-\delta/k_B T)$, where T is the current temperature and k_B the physical Boltzmann constant. When a large number of iterations is carried out keeping the same temperature, the system reaches thermal equilibrium. It is known that the probability distribution of the system states follows a Boltzmann distribution at thermal equilibrium. To maintain consistency, the chosen function for the acceptance criterion in the Metropolis algorithm also ensures that the system evolves into the Boltzmann distribution.

In the analogy, the different states of the system corresponds to the different feasible solutions to the Combinatorial Optimization problem, and the free energy of the system correspond to the function to optimize. Moreover, if the solid evolves into a metastable, it could be identified as a local optimal solution but, if in its evolution there are no defects, the solid has become a perfect crystal, which is identified as a global optimal solution for the combinatorial optimization problem. Nevertheless, this analogy is not complete since in the annealing process there is a physical variable which is the temperature in each state. In the algorithm routine, the "temperature" is simply a parameter that must be properly determined to get good solutions.

Nowadays, the term *Simulated Annealing (SA)* refers to the metaheuristic technique used to solve combinatorial problems.

In a theoretical aspect, the SA algorithm can be modelled as a sequence of Markov chains of finite length. That is, the probability of accepting a new feasible solution only depends on the previous accepted solution. However, this lack of memory can be fixed by storing the best solution so far. At the end, a descent algorithm can be used to get the local optimal solution generated in the earlier phases.

In subsections 4.3.1, 4.3.2 and 4.3.3 the algorithmic routine of SA and some specific features of this metaheuristic are described. At the same time, we detail the SA that we have designed for the (MC) problem. Then, in Subsection 4.6.3, a computational study is dedicated to determine the optimal combination of its parameters, using random instances.

4.3.1 Algorithmic routine of SA

Let S be the set of all feasible solutions and f be a cost function defined on elements of S. We deal with the problem of finding the solution/state $s_i \in S$ which maximizes f.

Algorithm 6: Local search

```
Require: An initial solution/state $s_{initial}$ from the solution space $S$.
$s_{current} = s_{initial}$
while $Stopping\ criteria = False$ do
    Compute $S(s_{current})$
    Select $s_{new} \in S(s_{current})$
    Compute the difference $\delta = f(s_{new}) - f(s_{current})$
    if $\delta > 0$ then
        $s_{current} := s_{new}$
    end
    Evaluate Stopping criteria
end
return An state $s_{current}$.
```

The SA is a type of Local Search algorithm (*descent* algorithm). A simple form of Local Search algorithm starts with an initial feasible solution $s_{initial}$ from the solution space S. This initial solution can be chosen from S in a deterministic or random manner. Then, at each iteration of the algorithm, a neighbor solution of the current one, $s_{new} \in S(s_{current})$, is generated by a suitable routine, being $S(s_{current})$ the neighborhood of $s_{current}$. If the objective value of this new feasible solution is better than the previous one, then the current solution is replaced. The process is repeated until no further improvement can be found in the neighborhood of the current solution and so the descent algorithm terminates at a local optimal solution (see Algorithm 6). This can be a disadvantage since the local optimal solution could be far from any global optimal solution. In SA, to avoid becoming trapped in a local optimum, the strategy is modified to accept sometimes a neighbor solution worse than the current one. That is, at each iteration, the probability of accepting a new solution is set as

$$P_T(\text{Accept } s_{new}) = \begin{cases} 1, & \text{if } f(s_{new}) > f(s_{current}), \\ -\exp\left(\frac{f(s_{current})-f(s_{new})}{T}\right), & \text{if } f(s_{new}) \leq f(s_{current}), \end{cases} \tag{4.1}$$

being T the current temperature. To maintain the analogy, the SA algorithm starts with a high value of T and it is gradually dropped after a certain number of neighborhood moves $n_{iter\_int\_max}$. Algorithm 7 describes the routine of SA.

Note that observing the acceptance criteria function (4.1) when $f(s_{new}) \leq f(s_{current})$, it follows that the smaller the decrease in the value of f, the more likely it is that the new solution generated will be accepted. Furthermore, the larger T, the more likely it is to accept worse solutions since T approaches zero. Besides, the larger $n_{iter\_int\_max}$, the more likely it is to reach good solutions. If both parameters $n_{iter\_int\_max}$ and T are short it is likely to be trapped in a local optimum. In the last iterations of the algorithm, it is very unlikely to accept worse solutions.

Concerning our purpose of getting good solutions for the maximization problem *(MC)*, the solution obtained in iteration k will be better than that of the previous iteration if it satisfies more demand than the prior one.

From the algorithmic point of view, there are some decisions to be made on which the behaviour of the SA hardly depends. On the one hand, the parameters of the cooling schedule have to be determined: the initial temperature, the reduction factor function, the number of iterations at each temperature and the stopping criteria. On the other hand, the neighborhood of any solution as well as the procedure to generate an initial solution must be defined.

Algorithm 7: Simulated Annealing

```
Require: Values for the initial temperature $T_{initial}$, the reduction factor $\alpha$,
the number of iterations at each temperature, $n_{iter\_int\_max}$, and a given
feasible initial solution $s_{initial}$.
$s_{current} = s_{initial}$
while Stopping criteria = False do
    $n_{iterations\_int} = 0$
    while $n_{iterations\_int} \leq n_{iter\_int\_max}$ do
        Compute $S(s_{current})$
        Select $s_{new} \in S(s_{current})$
        Compute the difference $\delta = f(s_{new}) - f(s_{current})$
        if $\delta > 0$ then
            $s_{current} := s_{new}$
        else
            Select random$(0, 1)$
            if random(0,1)< $\exp(-\delta/T)$ then
                $s_{current} := s_{new}$
            end
        end
        Update $n_{iterations\_int}$
    end
    Update temperature $T$
    Evaluate Stopping criteria
end
return The state $s_{current}$.
```

4.3.2 Cooling schedule

The *cooling schedule* is the set of parameters which greatly controls the strategy of SA: the initial temperature, the reduction factor function, the number of iterations $n_{iter\_int\_max}$ at each temperature and the stopping criteria. With respect to the temperature reduction function, a wide range has been proposed in the literature. For this work, a proportional temperature function is used. Then, at iteration k of the algorithm, the value of the temperature is

$$T^k = a\,T^{(k-1)}, \tag{4.2}$$

being $a \in (0, 1)$ a constant called the *reduction factor*. It is the rate at which the temperature is reduced. For the stopping criteria, a maximum number of outer iterations that attend to something or a temperature to be reached, among others, can be set. In our case, the stopping criteria is related to a maximum number of outer iterations, n_{iter_max}. The value given to each of these parameters is based on the idea of thermal equilibrium.

4.3.3 Neighborhood structure

The neighborhood structure is an aspect that highly reverberates on the quality of the final solution that is obtained. In this work, for (MC), the following neighborhood phase has been computed. Given a network, a neighboring network is built by eliminating the edge that generates the least loss in the objective function if it is not considered. Then, randomly, you continue adding edges of the remaining ones, allowing at most an addition of two. In this case, a network solution differs from its neighbor by at most three edges.

4.4 Adaptive Large Neighborhood Search

Adaptive Large Neighborhood Search (ALNS) is a term introduced by Stefan Ropke and David Pisinger in Ropke and Pisinger (2006). The ALNS concept extends the Large Neighborhood Search heuristic of Shaw (1997) which was based on ideas similar to those of the *ruin and recreate* method of Schrimpf et al. (2000). That is, ALNS refers to a Local Search algorithm, just like the SA. A difference of the ALNS exposed in Ropke and Pisinger (2006) with respect to the traditional Local Search algorithm is in the acceptance criteria of solutions. In the first, the acceptance criteria is the same as in the SA algorithm, instead of using a simple descent approach. Anyway, any other criterion could have been chosen in order to improve the solution obtained with the traditional Local Search algorithm. Furthermore, another feature of ALNS is that it considers a larger neighborhood, with respect to the size of the input data. The main difference between Shaw (1997) and Ropke and Pisinger (2006) concerns the way to construct the neighborhood. The routine of Ropke and Pisinger (2006) composes a neighborhood by a set of sub-heuristics that are competing with each other so that they are executed according to a frequency that refers to their historical performance.

As explained in Ahuja et al. (2002), for large problem instances, it is impractical to search explicitly each of the elements that make up the neighborhood. Thus, one must examine only a relatively small portion of the neighborhood or develop efficient algorithms for searching the whole neighborhood implicitly. The ALNS fits into the former category.

From the theoretical aspect, as with the SA algorithm, the ALNS can be modelled as a sequence of Markov chains of finite length. That is, the probability of accepting a new feasible solution only depends on the previous accepted solution. Moreover, since the acceptance criteria is the same as in SA, the solution from the last iteration is not necessarily the best one. However, this lack of memory can be fixed by storing the best solution so far. Eventually, a descent algorithm can be used to get the local optimal solution generated in the earlier phases.

In this section it is proposed an ALNS algorithm for the (MC) problem, using slightly some ideas of Ropke and Pisinger (2006) and Coelho et al. (2012). The specific features of the ALNS algorithm routine are explained in the following subsections. At the same time, we describe the ALNS that we have developed for the (MC) problem. Finally, in Subsection 4.6.4, a detailed study is dedicated to determine the optimal combination of its parameters.

4.4.1 Algorithmic routine of ALNS

A solution obtained from an ALNS algorithm is conditioned by its main algorithmic components: the way the neighborhood is built, the range of adaptive operators that interfere in the construction of solutions and the weight assigned to each one. The acceptance criteria of solutions also have an influence. Each of these factors is detailed below. At the same time, we exemplify the ALNS routine that we have developed for the (MC) problem, depicted in Algorithm 8.

Large neighborhood The number of nodes and/or edges in the underlying graph usually determines the size of the neighborhood. In our case, we have stated that two solution graphs s_i and s_j are considered neighbors if they have at most $1 + 2\,n_{del}$ different edges, being n_{del} the number of edges removed in the first phase of the selected operator. In Subsection 4.4.2, we explain how to generate a neighbor of a given graph.

Adaptive selection of operators At each iteration, alternating between different heuristics to modify and improve the current solution graph could give us a more robust heuristic overall, see Ropke and Pisinger (2006).

To select the heuristic to use, a weight is assigned to each one and a *roulette wheel selection principle* is executed. In this way, the proposed heuristics make up the operators of the roulette wheel mechanism. If there are k operators with weights w_k, $k \in \{1, 2 \ldots, k\}$, the probability of selecting the operator j is

$$\frac{w_j}{\sum\limits_{i=1}^{k} w_i}. \tag{4.3}$$

In the next subsection, we describe four sub-heuristics of the ALNS proposed for (MC).

Algorithm 8: Adaptive Large Neighborhood Search + GRASP

```
Require: Values for reduction factor $\alpha$, the reaction factor $\kappa$, the number
of segments $s$, the weights and the scores for the operators.
$n_{iterations} = 0$
$adjusted = False$
while $adjusted = False$ do
    Compute two initial solutions $s^1_{initial}$, $s^2_{initial}$ with GRASP
    Set $f_{max} = \max\{f(s^1_{initial}), f(s^2_{initial})\}$ and
     $f_{min} = \min\{f(s^1_{initial}), f(s^2_{initial})\}$
    if $f_{max} - f_{min} < \rho\% f_{max}$ then
        $\mathrm{T} = -\exp(f_{max} - f_{min}) / \ln(0.5)$
        $adjusted = True$
        $s_{current} = s^i_{initial}$ such that $f(s^i_{initial}) = f_{min}$
    end
end
while $Stopping\ criteria = False$ do
    $n_{iterations} = n_{iterations} + 1$
    Select an operator $j$ according to $w_j$, $j = 1, \ldots, 4$
    while the selected operator cannot be applied do
        Select an operator $j$ according to $w_j$, $j = 1, \ldots, 4$
        Attempt to apply operator $j$ to $s_{current}$
    end
    $s_{new}$ is the new solution graph obtained after applying operator $j$ to
     $s_{current}$
    Update $\theta_j = \theta_j + 1$
    if $f(s_{new} > f(s_{current}))$ then
        $s_{current} = s_{new}$
        if $f(s_{current} > f_{max})$ then
            $s_{best} = s_{current}$
            Update the score: $\pi_j = \pi_j + \sigma_1$
        else
            Update the score: $\pi_j = \pi_j + \sigma_2$
        end
    else
        if $s_{new}$ is accepted by the SA criteria then
            $s_{current} = s_{new}$
            Update the score: $\pi_j = \pi_j + \sigma_3$
        end
        Update temperature $T$
    end
    if $n_{iterations}$ is multiple of $s$ then
        Update the weights of all operators and reset their scores
    end
    Evaluate Stopping criteria
end
return The state $s_{current}$.
```

Adaptive weight adjustment It is possible to set the weights w_i, $\{1, 2, \ldots, k\}$ by hand, although it seems more natural to adjust them automatically at each iteration using statistics from the previous one. Hence, at the beginning of the ALNS all weights are fixed to 1 and all operators have the same probability of being selected. After each iteration, the score of each operator may be increased by σ_1, σ_2 or σ_3, which measures how well the operator applied has performed in this iteration. For it, the total number of iterations of the ALNS procedure n_{iter_max} is divided into a number of *segments* s. At the start of each segment, the score of each operator is set to zero. If it turns out that the operator applied in an iteration has resulted in a new global best graph solution stored so far, then its associated score is increased by σ_1. If simply the solution is better than the incumbent one, then its associated score is increased by σ_2. If it turns out that the new solution is worse than the current one but it has been accepted, then its score is increased by σ_3. It is natural to establish $\sigma_1 \geq \sigma_2 \geq \sigma_3 > 0$. That is, the better the solution, the more its score increases. At the end of each segment, the weights are updated using the recorded scores. Concretely, the weight associated to an operator j in segment $h+1$ is obtained as

$$w_{j,h+1} = \begin{cases} w_{j,h} & \text{if } \theta_j = 0, \\ (1-\kappa)w_{j,h} + \kappa\frac{\pi_j}{\theta_j} & \text{if } \theta_j \neq 0, \end{cases} \tag{4.4}$$

where π_j is the score of the operator j calculated during the last segment and θ_i is the number of times that the operator j has been used in this segment. Besides, the *reaction factor* κ controls how quickly the weight adjustment reacts to changes in the operator performance.

Acceptance and stopping criteria With the aim of avoiding getting trapped in a local optimum, not only solutions that are better than the current one are accepted. For that, the acceptance criteria function (4.1) of SA is adopted. The temperature starts out at an initial value and it is decreased every iteration using a function. This initial temperature T can be set by means of solutions obtained by the algorithm or fixed by a chosen value. In our proposal, the function is established to the proportional expression (4.2). A good choice of the initial temperature T may be set by inspecting two initial solutions $s^1_{initial}$ and $s^2_{initial}$. Let us denote by $s^{max}_{initial}$ the one with the largest objective value and by $s^{min}_{initial}$ the one with the smallest objective value. The start temperature is now set such that if $s^{min}_{initial}$ solution is $\rho\%$ worse than $s^{max}_{initial}$, that is $f(s^{max}_{initial}) - f(s^{min}_{initial}) > \rho\% s^{max}_{initial}$, then $s^{min}_{initial}$ is accepted with probability 0.5:

$$P_T(\text{Accept } s^{min}_{initial}) = \exp\left(\frac{f(s^{max}_{initial}) - f(s^{min}_{initial})}{T}\right) = 0.5 \tag{4.5}$$

or, equivalently,

$$T = -(f(s^{min}_{initial}) - f(s^{max}_{initial}))/\ln(0.5). \tag{4.6}$$

The parameter $\rho\%$ is selected by the user so that the routine does not take a long time in the step of calculating the initial temperature T. With respect to the stopping criteria, a maximum number of iterations n_{iter_max} has been set.

4.4.2 Neighborhood structure and Operators

While the SA only uses a heuristic to build the neighborhood, the ALNS uses the entire set of operators to build neighboring solutions to the current one. In each iteration of the algorithm, the operators construct a new neighbor.

The proposed ALNS for (MC) considers four *compound operators*, defined in the following way. Each of them is composed of two *phases* or *simple operators.* The simple operators are classified according to two types: *repair operators* and *destroy operators.* The first phase of each compound operator consists of applying a destroy operator and then, in the second phase, a repair operator is executed. That is, firstly, n_{del} edges are removed from the current solution graph and then, at most $1 + n_{del}$ edges are added to try to improve the current network.

The destroy operators considered are:

- Deleting the minimum worst n_{del} edges. This operator selects the edge that causes the smallest leak in the covered demand of the current solution graph if it is deleted from it. Once this edge is identified, it is definitely removed. Then, the graph is updated and $n_{del} - 1$ edges are removed from it in this way.
- Deleting n_{del} random edges. This operator randomly selects an edge from the current solution graph to remove it. Then, the graph is updated and $n_{del} - 1$ edges are removed from it in this way.

The repair operators considered are:

- Adding the $1 + n_{del}$ remaining best edges at most. This operator selects the remaining feasible edge that causes the largest gain in the demand to cover if it is added to the current solution graph. Once it is identified, it is definitely added. Then, the graph is updated, and also the remaining budget. If it is possible, more edges are added in this way, until adding $1 + n_{del}$ at most.
- Adding $1 + n_{del}$ random remaining edges at most. This operator randomly selects edges from the remaining set of feasible edges to add them to the current solution graph in a sequential manner. First, randomly a remaining feasible edge is selected and added to the current solution graph. Then, the available

budget is updated and, if it is possible, more edges are added in the same way, allowing $1 + n_{del}$ new edges at most.

Thus, the four compound operators are:

- Deleting the n_{del} worst edges - adding the $1 + n_{del}$ remaining best edges
- Deleting the n_{del} worst edges - adding $1 + n_{del}$ random remaining edges
- Deleting n_{del} random edges - adding the $1 + n_{del}$ remaining best edges
- Deleting n_{del} random edges - adding $1 + n_{del}$ random remaining edges

4.5 Genetic Algorithms

Genetic algorithms (GA) were developed by Holland (1975) and were later modified by other researchers in order to improve them. The goal was to abstract and rigorously explain the adaptive processes of natural systems of genetics to design artificial systems software.

Genetic algorithms are search and optimization procedures based on the mechanics of the principles of natural genetics and selection. They combine survival of the fittest among a population of individuals with structured yet randomized information exchange. This exchange is governed by operators that simulate the phases of selection, crossover and mutation that occur in natural genetics.

In natural genetics, a number of individuals, namely the *population*, evolves through several *generations* due to the genetic evolution process. Each *individual* is identified by its *chromosome*, which contains a number of *genes*. The possible values that genes can take are the *alleles*. That is, the chromosome is coded as a vector of fixed length (*string*) with values from the alleles. This complete gene encoding is called the *genotype* of the individual. Throughout this natural procedure, the most adapted individuals survive, and the less adapted are discarded. In the analogy, the population corresponds to a set of feasible solutions generated by a different technique. This set of feasible solutions evolves through several iterations of the algorithm. The chromosome of an individual represents a set of variables, i.e., a solution. To encode a solution one has to identify the kind of genotype the problem needs. Besides, to evaluate a solution, a fitness function is defined according to the problem at hand. Nevertheless, just as it happens with the SA algorithm, this analogy is not complete. In the routine of a genetic algorithm, some parameters are used to best simulate the natural genetic operators and thus be able to discard the least fit solutions.

In this section, we will briefly describe the general mechanics of a genetic algorithm. At the same time, we describe the one developed in Perea et al. (2020) for

(MC) problem. Then, in Subsection 4.6.5, we explain some computational conclusions about the GA of Perea et al. (2020). After, we test it using a large instance.

4.5.1 Algorithmic routine of GAs

Algorithm 9 describes the general procedure of a genetic algorithm. For that, an initial set of feasible solutions $|S|$ is considered, which can be created in a deterministic or random manner. Then, at each iteration, a series of basic operators, explained in Subsection 4.5.2, are applied in the following order. First, the selection operator chooses two individuals (solutions) from the current set of solutions, s_i and s_j. Next, these two elements are the protagonists of the crossover phase to form two new individuals, named the resulting *offspring*. They will replace other elements of the population according to some criteria established in relation to the fitness function. Taking into account this update in the population, the mutation operator is executed to select some individuals of the current population to flip randomly some positions in their strings. Finally, the population is updated again. This routine is repeated until the stopping criteria allows it. The stopping criteria is common to be related to a maximum number of iterations, a maximum execution time of the algorithm or the achievement of a solution with a sufficiently good value of the fitness function among others. Eventually, the best solution of the final state for the population represents the solution of the algorithm.

Algorithm 9: Genetic Algorithm

```
Require: Values for the probability $p_m$ of executing the mutation operator
and a given initial population of feasible solution graphs.
Encoding the population
while Stopping criteria = False do
    Perform crossover operator with probability $p_c$
    Perform mutation operator with probability $p_m$
    Evaluate Stopping Criteria
end
return The best element of the current population.
```

The main components of a genetic algorithm are described bellow.

Encoding In order to apply a GA to a given problem, the first decision to be made is the kind of genotype to select. That is, to identify the encoding the problem needs. There are several types of encodings. For the (MC) problem, the solution is characterized by the set of edges that are built from the underlying graph $G = (N, E)$. Thus, the chosen encoding by Perea et al. (2020) uses a vector of length $|E|$ for which each component has value 1 if the edge associated with that position appears in the solution network, and 0 otherwise.

The fitness function For a given problem, a fitness function has to be defined to measure the quality of any potential solution. In Perea et al. (2020), the fitness function considered only counts the amount of demand served.

Initial population A set of individuals is needed to apply a GA. These individuals must be solutions or parts of solutions to the problem to optimize. In Perea et al. (2020), the individuals of the initial population to be considered are built by computing the constructive phase of the GRASP described in Subsection 4.2.

4.5.2 Neighborhood structure and Operators

SA and ALNS use a probability function that allows the acceptance of a worse solution with a decreasing probability as the search progresses. With GAs, a pool of solutions is used, and the neighborhood is composed of the interaction of pairs of solutions in the execution phase of the crossover operator and also by the disturbance of some individuals in the execution phase of the mutation operator. Note that if none of the crossover and mutation operators is executed, the final solution of a GA is the same as the initial one. That is, the best element of the initial population coincides with the final solution.

With regard to operators, the following are the most frequent.

Selection Each solution of the population has an associated probability that tries to simulate the reproductive ability of an individual according to natural genetics. Normally this probability is related to the value of the fitness function that has such solution. This operator can be understood as the initial step for the crossover and mutation operators, which selects a required number of solutions. Authors in Perea et al. (2020) set the same probability for each solution for being selected.

Crossover With this operator, two selected strings of the population exchange one or more parts from some chosen positions, in a random or deterministic manner, to create two new solutions. Then, these two offspring are evaluated by means of some criteria. The fitness function is used to replace or not other solutions of the population by these two offspring. This phase may have an associated probability of execution, p_c. Note that, the crossover operator can be executed only in a pair of solutions or consider more pairs of solutions to exchange. Authors in Perea et al. (2020) set $p_c = 1$, i.e. the crossover phase is always executed and only to one pair of solutions. Besides, two types of crossover operators have been performed. To understand them, let us consider the following two subnetworks of such a given underlying network, $\mathcal{N}_1 = (\{1,2,3\}, \{\{1,2\},\{2,3\}\})$ and $\mathcal{N}_2 = (\{1,2,3\}, \{\{1,3\},\{2,3\}\})$, which are coded as $s_1 = (1,0,1,0)$ and $s_2 = (0,1,1,0)$, respectively.

- Single-point crossover. An integer $1 \leq k \leq |E| - 1$ is randomly generated. This integer separates each string into two parts. Two offspring, s_1 and s_2 are generated as follows. On the one hand, the part formed by the first k elements of s_1 is joined together with the last $|E| - k$ elements of s_2. Conversely, on the other hand, the part formed by the first k elements of s_2 is joined together with the last $|E| - k$ elements of s_1. That is, assume the coded solutions $s_1 = (1,0,1,0)$ and $s_2 = (0,1,1,1)$. If $k = 1$, the new two offspring are $s_1' = (1,1,1,1)$ and $s_2' = (0,0,1,0)$.

- Two-point crossover. Two integers $1 \leq k_1 \leq k_2 \leq |E| - 1$ are randomly generated. These integers separate each string in three parts. Two offspring, s_1 and s_2 are generated as follows. On the one hand, the part formed by the first k_1 elements of s_1, the part formed by the k_1 element to the k_2 element of s_2 and the part formed by the last $|E| - k_2$ elements of s_1 are joined. Conversely, on the other hand, the part formed by the first k_1 elements of s_2, the part formed by the k_1 element to the k_1 element of s_2 and the part formed by the last $|E| - k_2$ elements of s_2 are joined. That is, assume the coded solutions $s_1 = (1,0,1,0)$ and $s_2 = (0,1,1,1)$. If $k_1 = 1$ and $k_2 = 3$, the new two offspring are $s_1' = (1,1,1,0)$ and $s_2' = (0,0,1,1)$.

Each of the two offspring may be characterized by one of the following two cases. On the one hand, if the cost of the resulting offspring is larger than the available budget, then it is an infeasible solution graph for the problem. This solution is corrected by removing the edge that most reduces the total cost. This procedure is repeated until the total cost is less than or equal to the available budget. On the other hand, it could be that the cost of the resulting offspring was lower than the available budget. In this case, if there are remaining edges that can be built, these are added in the same way as in GRASP (see Section 4.2). Once the offspring is corrected or extended, it is added to the current population if its value of the fitness function induces an improvement with respect to the worst solution of the current population, which is removed.

Mutation In this phase, some solutions are selected from the population according to the selection operator. Then, in each of them, some positions are chosen in their associated strings to flip them or exchange them in some way. These positions can be chosen in a random manner or according to an associated probability to each of them. Just like the crossover phase, the mutation operator has a probability related to its execution, p_m. In Perea et al. (2020), the mutation operator only selects one solution from the current population randomly. Then, a random position with value 1 and a random position of value 0 are chosen and they are exchanged with a certain probability p_m. If the cost of the resulting solution is larger than the available budget (infeasible solution graph) or there is budget left to continue adding edges to the solution, it is corrected or extended following the same procedures defined before for the crossover operator.

4.6 Computational experiments

In this section, it is shown that the performance of the proposed metaheuristics algorithms of Sections 4.3, 4.4 and 4.5 is influenced by the values that their parameters take. In what follows, we will refer as GRASP, SA, ALNS and GA to the different metaheuristics previously described for the (MC) problem. The purpose is to measure the quality of the solutions they obtain. Note that, for each algorithm, some of the involved parameters are continuous and belong to a very large range. Then, we cannot try all combinations of all of them for any instance given. Furthermore, even if one could find the optimal parameters combination at a specific instance, it might not be optimal for another different instance given. However, it is also expected that a specific combination of the input parameters behaves in a similar way for different-sized networks but with similar topology.

To carry out the proposed objective, first Subsection 4.6.1 shows a detailed description of the two groups of instances used to test such metaheuristics. Then, in Subsection 4.6.2, the best configuration of the GRASP is identified with the aim of using it for the next experiments as a generator of initial solution networks. Subsections 4.6.3, 4.6.4 and 4.6.5 contain an extensive sensitivity analysis for SA, ALNS and GA, using the random instances sets N20 and N80 described in Subsection 4.6.1. These experiences are used to analyze the behavior of each metaheuristic, not to compare them with each other. After identifying the best configuration for each of them, they are executed using a large instance to compare them.

Regarding the sensitivity analysis of the improvement algorithms, the explored variables are the following. For the sensitivity analysis of GRASP, only the first two have been used.

- `Value`: On the one hand, for the computational results related to the GRASP, this variable measures the overall normalized objective value of the solution. We divide the objective value obtained with the GRASP by the objective value in the situation where nothing is built. On the other hand, for the rest of the metaheuristic algorithms, this variable measures the increase in coverage from the GRASP solution.

- t: The overall execution time.
- RPD: The average relative percent deviation with respect to the best solution found by the BD_CW implementation after 1 hour. It is computed by the following formula:
$$\texttt{RPD} = 100\frac{Z_{\texttt{BD\_CW}} - Z'}{Z_{\texttt{BD\_CW}}}, \tag{4.7}$$
being Z' and $Z_{\texttt{BD\_CW}}$ the values of the solutions obtained by the GRASP and the BD_CW implementation, respectively.

Besides, for the GA, one more variable is also studied:

- Population_Value: the overall percentage increase in the population average coverage.

So that the results between the different instances are comparable, the objective value is normalized by G_{total}. The variables Value, RPD, and Population_Value are constructed with this normalized value.

All of the instances in N20 were solved to optimality, thus $Z_{\texttt{BD\_CW}}$ represents the optimal solution value in this case. For the instances in N80, none of them was solved to optimality in 1 hour, then $Z_{\texttt{BD\_CW}}$ represents the objective value of the best solution found. Note that for the case in which $Z_{\texttt{BD\_CW}}$ represents the optimal solution, the smaller the value of RPD, the better the quality of the solution found by the metaheuristic. However, if $Z_{\texttt{BD\_CW}}$ does not correspond with the optimal solution and $Z' > Z_{\texttt{BD\_CW}}$, the greater the RPD, the better the solution found by the metaheuristic is.

All our computational experiments were performed on a computer equipped with an Intel Core i5-7300 CPU processor, with 2.50 gigahertz 4-core, and 16 gigabytes of RAM. The operating system is 64-bit Windows 10. Codes were implemented in Python 3.8. These experiments have been carried out through CPLEX 20.1 solver, named CPLEX, using its Python interface. CPLEX parameters were set to their default values and the models were optimized in a single-threaded mode. Besides, the different statistical analyses have been done through the use of R.

Due to the large dimensions of the collected data for this analysis, we have added it as supplementary material in https://github.com/Natividad13/GRASP_for_Network_Design, https://github.com/Natividad13/SA_for_Network_Design, https://github.com/Natividad13/ALNS_for_Network_Design and https://github.com/Natividad13/GA_for_Network_Design.

4.6.1 Data sets: random instances

The random instances used to test the metaheuristic described are generated in the same manner as in Subsection 2.4.1 with some different points. We have considered networks of size $n \in \{20, 40, 60, 80\}$ nodes. We name these instances as N20, N40, N60 and N80. There are 20 underlying networks of each size. The underlying planar graph of each instance has been generated with higher density. That is, their edges are deleted with a probability of 0.2. The budget αC_{total} is equal to 40% of the cost of building the whole underlying graph considered, which means that $\alpha = 0.4$. Besides, to build the set of O/D pairs W all possible ones are taken into account. Thus, set W is composed of $n(n-1)$ elements. Parameter u^w is set to 2 times the Euclidean length between w^s and w^t. The rest of the parameters of the instance are chosen as in Chapter 2.

For the final experiment, we have used a network of size $n = 100$ nodes, generated randomly and in the same way as the others.

4.6.2 Preliminary experiments for GRASP

With the purpose to analyze the performance of the GRASP when changing its parameters k and n_{iter_max}, a sensitivity analysis is done using the sets of random instances N20, N40 and N60. For that, all the possible combinations of the following chosen values for each of both parameters are tested, being $\alpha = 0.4$.

- The stopping criteria: $n_{iter\_max} \in \{10, 20, 40, 100\}$
- Cardinality of the set of best edges to select from:
  $k \in \{n/7, n/5, n/4, n/3\}$

		k							
	n_{iter_max}	$n/7$		$n/5$		$n/4$		$n/3$	
		t	Value	t	Value	t	Value	t	Value
N20	10	6.377	0.340	5.625	0.320	5.576	0.323	5.579	0.313
	20	11.311	0.339	10.962	0.339	11.170	0.333	10.885	0.328
	40	24.236	0.340	21.648	0.334	21.582	0.337	21.264	0.335
	100	54.884	0.346	54.080	0.345	53.773	0.347	52.166	0.346
N40	10	189.950	0.308	182.784	0.295	179.184	0.291	171.244	0.284
	20	351.891	0.313	347.342	0.312	329.162	0.300	318.899	0.292
	40	703.500	0.327	691.119	0.315	663.588	0.308	641.381	0.301
	100	1731.961	0.332	1689.035	0.328	1630.774	0.314	1570.210	0.303
N60	10	1301.410	0.292	1193.310	0.259	1224.868	0.264	1109.236	0.256
	20	2482.400	0.284	2334.754	0.279	2241.138	0.283	2107.454	0.273
	40	4971.832	0.302	4669.736	0.290	4436.286	0.286	4159.585	0.276

Table 4.1: Sensitivity analysis for the GRASP of (MC) using N20, N40 and N60 random instances.

In Table 4.1, we report variables `t` and `Value` for the sets of random instances. We have grouped these computational results by four different increasing values for n_{iter_max} and, fixing it, by four different values for parameter k.

For each set N20, N40 and N60, an ANOVA analysis is performed to measure whether there is significant evidence between the different values of n_{iter_max} and k for each of the variables. The individual p-values are shown in Table 4.2. With respect to the variable `Value`, for N40 and N60 instances there exists significant evidence to confirm that the different levels of these two parameters affect. For N20, there does not exist significant evidence to confirm that the different values of the parameters affect `Value`, at the 95% confidence level. Respecting variable `t`, there exists significant evidence to confirm that the different levels of n_{iter_max} affect it, using any of the three sets of instances N20, N40 and N60. There is no significant evidence to say that the interaction between n_{iter_max} and k affects it in any case. It seems that for medium size instances, it is significant the choice of the values of the parameters k and n_{iter_max}.

		`Value`	`t`
N20	n_{iter_max}	0.0706	<2e-16
	k	0.6356	0.715
N40	n_{iter_max}	0.000208	<2e-16
	k	0.000156	0.517
N60	n_{iter_max}	0.00548	< 2e-16
	k	0.01148	0.000919

Table 4.2: P-values of the ANOVA for the GRASP of (MC) using N20, N40 and N60 instances.

Figure 4.1 shows the results of columns `Value` in Table 4.1. By observing this figure and focusing on N40 and N60 instances, we can say that the larger is n_{iter_max} or the smaller is k, the larger the coverage. That is, the greater the number of generations fixed or the smaller the set of feasible remaining edges to select from, the more demand is covered. Then, for N40 and N60, the best options considered are $(n_{iter\_max}, k) = (100, n/7)$ and $(n_{iter\_max}, k) = (40, n/7)$, respectively. In consequence, for the experiments tested in the next subsections, k is fixed to $n/7$.

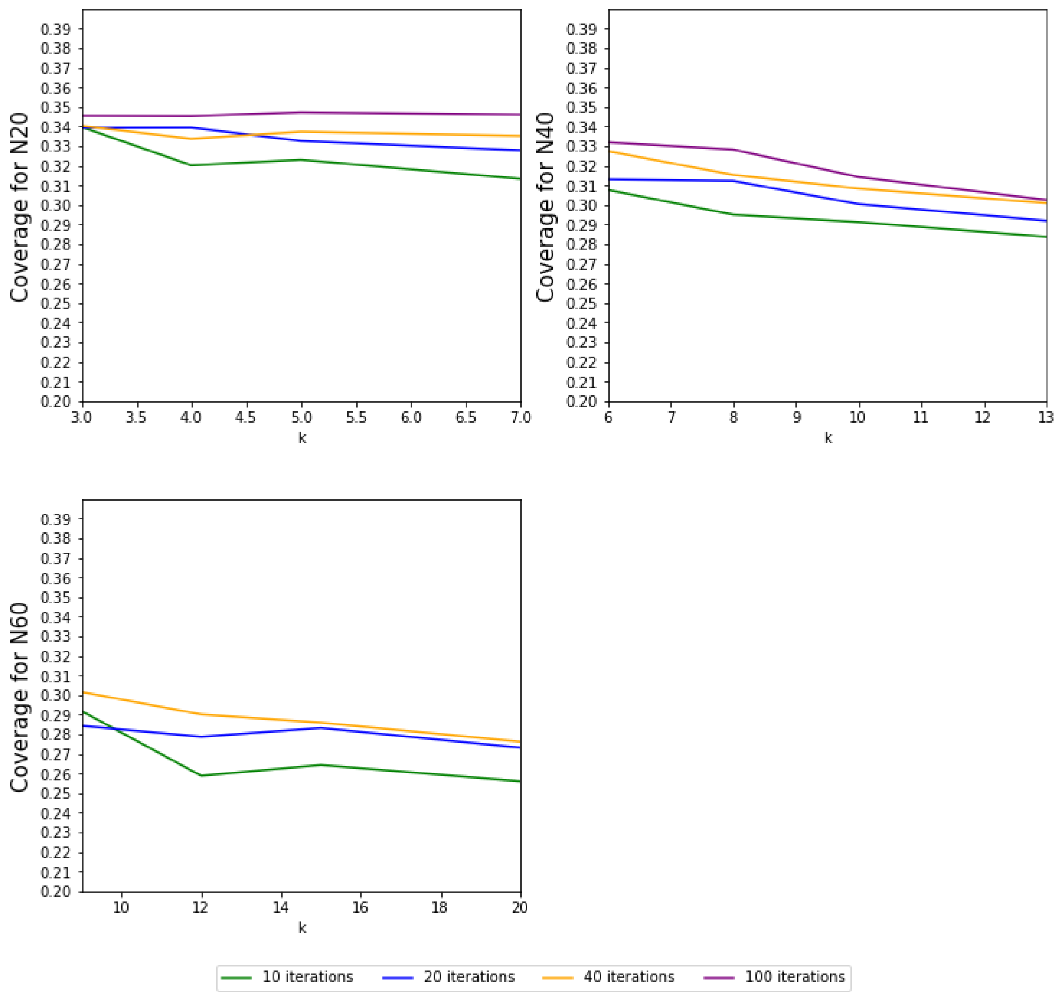


Figure 4.1: Sensitivity analysis for GRASP for (MC) using N20, N40 and N60 instances.

4.6.3 Preliminary experiments for SA

A sensitivity analysis for the SA algorithm proposed in Section 4.3 is done for N20 and N80 instances, being $\alpha = 0.4$. For N20, it is composed considering all the possible combinations of the following chosen values for each of the parameters.

- The stopping criteria: $n_{iter\_max} \in \{5, 10, 20, 40\}$
- Number of interior iterations: $n_{iter\_int\_max} \in \{5, 10, 20, 40\}$
- Initial temperature: $T \in \{100, 200, 500, 1000, 2000\}$
- The reduction factor (cooling rate factor):
  $r \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$

Therefore, each of the 20 random instances of N20 was solved by SA with each of the possible $4^2 \cdot 5 \cdot 9 = 720$ combinations of the parameters, having in total $720 \cdot 20 = 14400$ executions of SA for the (MC) problem. After analyzing this experiment, a similar one was done for N80. Due to the size of the networks N80 and from the conclusions of the first experiment, some different values for the parameters were tested for the second one.

- The stopping criteria: $n_{iter\_max} \in \{5, 10\}$
- Number of interior iterations: $n_{iter\_int\_max} \in \{5, 10, 20\}$
- Initial temperature: $T \in \{100, 200, 400, 800, 1600\}$
- The reduction factor (cooling rate factor): $r \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$

For N80, the possible combinations are 150, but those with $n_{iter\_max} = 10$ and $n_{iter\_int\_max} = 20$ were not taken into account due to the fact that their computational time is larger than 1 hour. Thus, 125 combinations were tested, having in total 2500 executions of SA. Computational data respecting this experiment are added as supplementary material in `https://github.com/Natividad13/SA_for_Network_Design`.

	`Value`	`t`	`RPD`
n_{iter_max}	0.204	<2e-16	0.429
$n_{iter\_int\_max}$	5.31e-05	<2e-16	6.43e-13
T	2.32e-09	0.044	< 2e-16
r	0.215	0.959	0.005

Table 4.3: P-values of the ANOVA for the SA of (MC) with N20 instances.

	`Value`	`t`	`RPD`
n_{iter_max}	0.951	1.36e-09	0.996
$n_{iter\_int\_max}$	0.029	< 2e-16	0.969
T	0.278	0.683	0.990
r	0.294	0.963	0.999

Table 4.4: P-values of the ANOVA for the SA of (MC) with N80 instances.

For each set of instances, an ANOVA analysis is performed to measure whether there is significant evidence between the different values of these parameters for each of the variables. The individual p-values are shown in Tables 4.3 and 4.4 for N20 and N80, respectively. For N20, we observe that there is no significant evidence to confirm that the different considered values of n_{iter_max} and r affect `Value` and that there is significant evidence to accept that the different considered values of T and $n_{iter\_int\_max}$ affect this variable. For N80, there is significant evidence to accept that the different considered values of $n_{iter\_int\_max}$ affect `Value`. There is no significant evidence to confirm that the different considered values of n_{iter_max}, T and r affect this variable. With respect to variable `t`, for both sets of instances, there is significant evidence to accept that the different levels of n_{iter_max} and $n_{iter\_int\_max}$ affect it. Besides, for N20 there is significant evidence to accept that the different levels of the temperature also affect `t`. By last, for `RPD`, there is significant evidence to accept that the different levels of $n_{iter\_int\_max}$, T and r affect it with the set of instances N20. The same cannot be obtained for N80, for which there is no evidence to accept that the different levels of the four factors considered affect `RPD`.

Increase in coverage Figures 4.2 and 4.3 show the mean values of variable `Value` for N20 and N80 considering the specified combinations of the parameters in each case. That is, each point of these line graphs has associated a 4-tuple $(n_{iter\_max}, n_{iter\_int\_max}, T, r)$.

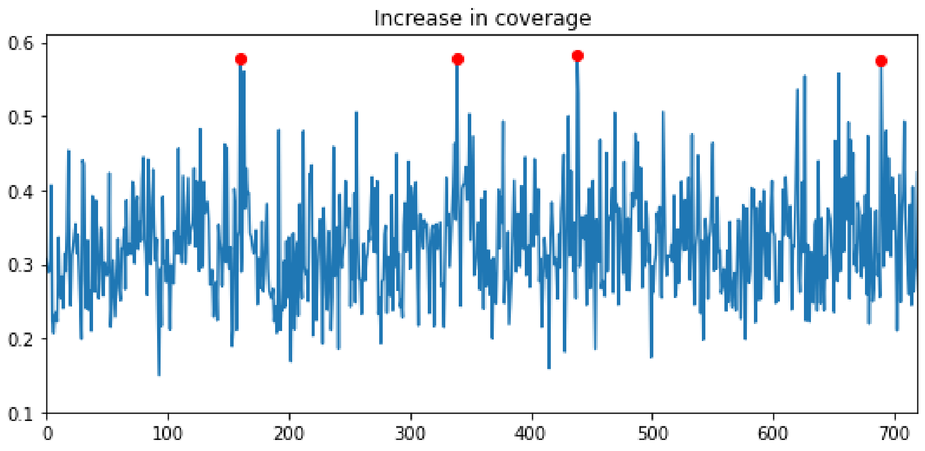


Figure 4.2: Mean values of the overall percentage increase in coverage for all the 720 combinations of the parameters of SA over the GRASP for (MC), using N20. Red points highlight the best configurations.

We can see that for both sets of instances N20 and N80, all the configurations improve the solution obtained from the GRASP, being the small improvement of 14.97% for N20 and 5.73% for N80. Intuitively, for N20, the best configurations of these parameters could be $(5, 40, 500, 0.8)$, $(10, 40, 500, 0.7)$, $(20, 10, 1000, 0.7)$ and $(40, 40, 200, 0.6)$, with an improvement of 57.88%, 57.77%, 58.28% and 57.49%, respectively. For N80, the best configurations are $(5, 10, 800, 0.6)$ and $(5, 20, 1600, 0.9)$, with an increase in coverage from the GRASP of 22.37% and 22.88%, respectively.

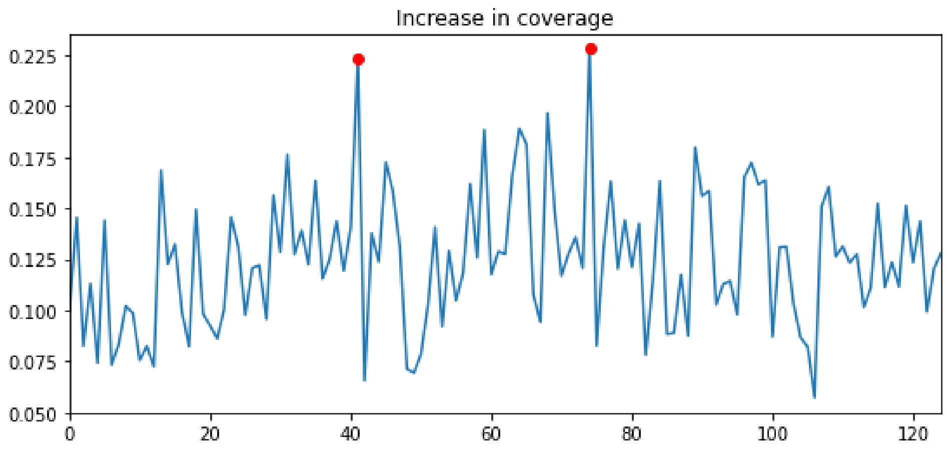


Figure 4.3: Mean values of the overall percentage increase in coverage for all the 125 combinations of the parameters of SA over the GRASP for (MC), using N80. Red points highlight the best configurations.

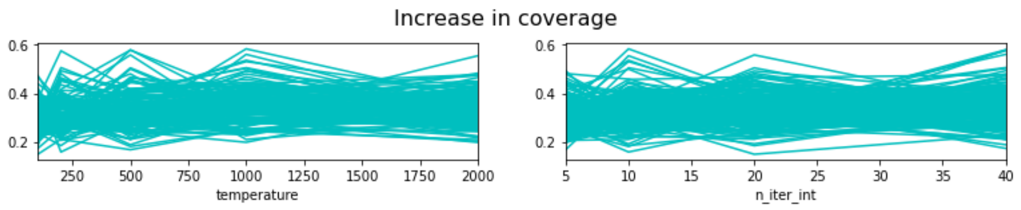


Figure 4.4: Mean values of the overall percentage increase in coverage according to the variation of the number of interior iterations and temperature parameters of SA over the GRASP for (MC) using N20.

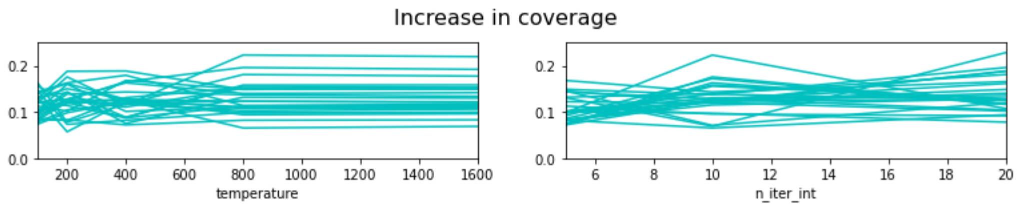


Figure 4.5: Mean values of the overall percentage increase in coverage according to the variation of the number of interior iterations and temperature parameters of SA over the GRASP for (MC) using N80.

With respect to obtaining some conclusions about the effect of the different levels of the parameters on the increase in coverage, some observations can be stated from Figures 4.4 and 4.5. Factors analyzed and drawn are the only ones that have evidence of affect in the previous ANOVAs executed. For N80, we observe that the increase in coverage is greater as the iterations and the temperature increase, for the majority of the cases. For N20, this feature does not seem to be so obvious.

Quality of solutions For N80, none of the instances was solved to optimality using the `BD_CW` routine. Observing the collected data for this study, if we compare the best solution found in 1 hour by the `BD_CW` routine with the solutions found by the different configurations of SA (whose time is less than 1 hour), it turns out that, for 59.6% of the cases, the metaheuristic achieves better results (considering the set of all instances used).

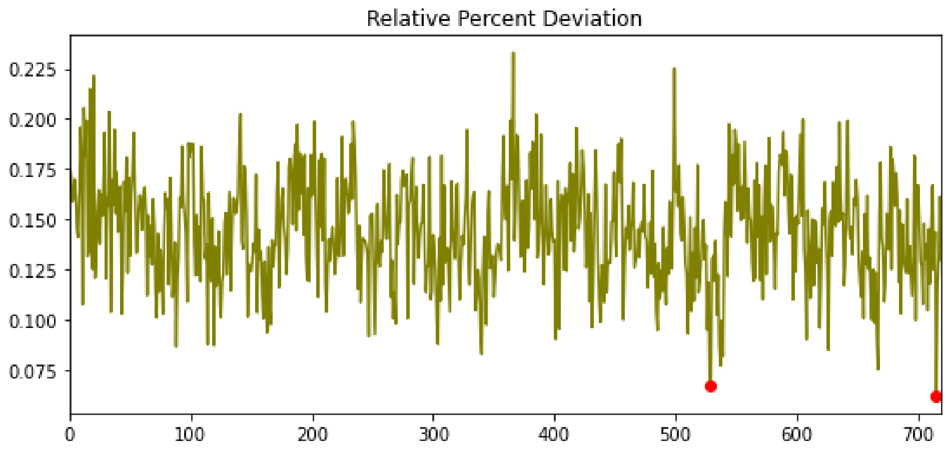


Figure 4.6: Mean values of the overall `RPD` for all the 720 combinations of parameters of SA for (MC) using N20. Red points highlight the best configurations.

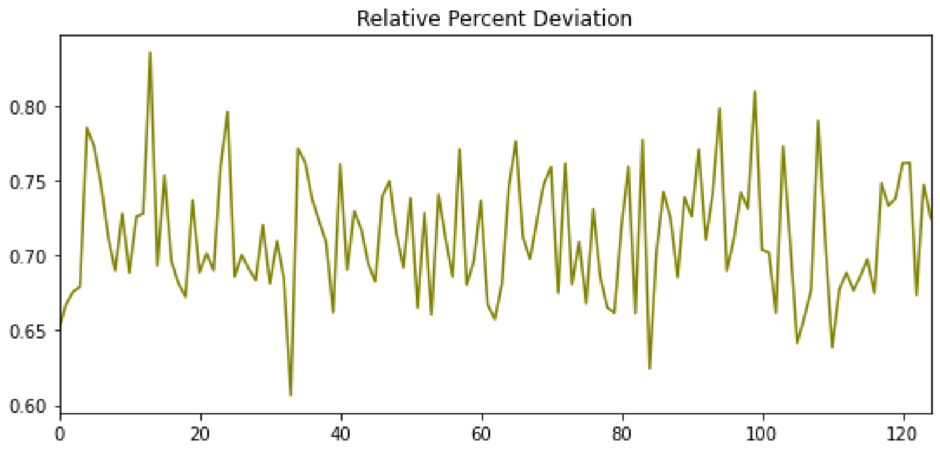


Figure 4.7: Mean values of the overall `RPD` for all the 125 combinations of parameters of SA for (MC) with N80.

In terms of deviation with respect to the solution obtained by the `BD_CW` routine, the SA algorithm with configurations $(40, 40, 2000, 0.5)$ and $(20, 40, 1000, 0.8)$ yields the best results for N20, as shown in Figure 4.6. These configurations find solutions only 6.19% and 6.68% different from the optimal solutions. The previous ANOVA described shows that there is significant evidence to accept that the different values of the parameters, except n_{iter_max}, affect RPD. Nevertheless, for N80, there is no significant evidence to accept that the different values considered for its case affect RPD. We observe in Figure 4.7 that the solutions found by SA are at least 60.65% different from the best solution found by the BD_CW routine after 1 hour. Working with the collected data, we get that this situation is positive for 59.6% of the cases since for them the solution found by SA is better than that reached by using the BD_CW routine during 1 hour. This situation is negative for 40.4% of the cases. Thus, we consider two sets. On the one hand, cases in which SA gets better results, being configuration $(10, 10, 800, 0.7)$ the best one. On the other hand, cases for which SA has got a solution worse than that achieved by the BD_CW routine. For this situation, $(5, 20, 400, 0.6)$ is the best configuration.

CPU time Finally, we analyze the computational effort of the SA algorithm presented. The average CPU time for all of the combinations tested are collected in the supplementary material. Combinations with a small value of $n_{iter\_int\_max}$ are those which present a significative short computational time compared with the rest of them. By fixing the value of $n_{iter\_int\_max}$, there is no significant difference in the computational time. For N20, the average CPU time of the BD_CW routine is $53.05s$. Roughly speaking, the SA algorithm gets solutions that are on average 6.4% (taking into account the best configurations) different from the MILP solution (see Figure 4.6) and in a fifth of the BD_CW routine time. The previous ANOVA described shows that n_{iter_max}, $n_{iter\_int\_max}$ and T significantly affect t. For N80, the BD_CW routine was executed for 1 hour. The average CPU time for all the combinations tested for the SA is shown. That is, the average CPU time of SA is $1502.41s$. Thus, in less than half an hour, for 59.6% of the cases, the SA algorithm gets solutions which are on average 71.26% better than the solution obtained by the BD_CW routine. However, in less than half an hour, for 40.4% of the cases, the SA algorithm gets solutions which are on average 71.26% worse than that of the BD_CW routine. As for N20, the previous ANOVA described shows that n_{iter_max}, $n_{iter\_int\_max}$ and T significantly affect t using N80 instances.

Summary of results Table 4.5 shows, for each of the variables studied, the best combinations of SA algorithm factors in N20 and N80 instances. As we noted before, not all the factors are individually significant for each variable studied.

Note that having a large increase in coverage may be due to the fact that the GRASP solution from which SA starts is not good. On the other hand, having a RPD close to 0 means that the solution provided by the metaheuristic routine is quite good. This may be due to the fact that the GRASP solution from which SA starts is also quite good and, in this case, the increase in coverage with respect to the

GRASP solution is small. This is exactly what happens for N20 if we observe Table 4.6. Anyway, for `Value` there is dominance for the combinations $(10, 40, 500, 0.7)$ and $(20, 10, 1000, 0.7)$ over the rest of them. They are characterized by a large value of T and a small value of $n_{iter\_int\_max}$, or vice versa. For `RPD`, the best configuration is $(40, 40, 2000, 0.6)$, which is featured by large values of n_{iter_max}, $n_{iter\_int\_max}$ and T. Similar conclusions are obtained for N80. Combinations characterized by these previous conclusions are the ones that will be tested later in Subsection 4.6.6 with a large instance.

Variable	Best combinations for N20	Best combinations for N80
`Value`	$(5, 40, 500, 0.8)$, $(10, 40, 500, 0.7)$, $(20, 10, 1000, 0.7)$, $(40, 40, 200, 0.6)$	$(5, 10, 800, 0.6)$, $(5, 20, 1600, 0.9)$
`t`	All the combinations with $n_{iter\_int\_max} = 5$	All the combinations with $n_{iter\_int\_max} = 5$
`RPD`	$(40, 40, 2000, 0.6)$, $(20, 40, 1000, 0.8)$	$(10, 10, 800, 0.7)$ if $Z_{SA} \geq Z_{\texttt{BD\_CW}}$, $(5, 20, 400, 0.6)$ if $Z_{SA} < Z_{\texttt{BD\_CW}}$

Table 4.5: Best combinations of SA algorithm factors.

Variable	Combination	`Value`	`RPD`
`Value`	$(5, 40, 500, 0.8)$	57.88%	10.04%
	$(10, 40, 500, 0.7)$	57.77%	9.16%
	$(20, 10, 1000, 0.7)$	58.28%	9.87%
	$(40, 40, 200, 0.6)$	57.49%	12.69%
`t`	All the combinations with $n_{iter\_int\_max} = 5$	around 30.69%	around 15.51%
`RPD`	$(40, 40, 2000, 0.6)$	40.45%	6.19%
	$(20, 40, 1000, 0.8)$	35.68%	6.68%

Table 4.6: Analysis of the best combinations of SA algorithm factors using N20.

4.6.4 Preliminary experiments for ALNS

The proposed ALNS algorithm 4.4 is controlled by quite a few parameters. To measure the quality of the solutions obtained by ALNS, we have conducted a sensitivity analysis for the sets of random instances N20 and N80. We have fixed $\alpha = 0.4$. For N20, all the possible combinations of the following values considered for each of the parameters are tried.

- Number of edges to delete to form neighbors: $n_{del} \in \{1, 2\}$
- The stopping criteria: $n_{iter\_max} \in \{5, 10, 20, 40\}$

- Length of a segment: $s \in \{2, 3, 5\}$
- The reaction factor: $\kappa \in \{0.25, 0.5, 0.75\}$
- Weight adjustment. Three configurations have been tested:
  - Configuration 1: $\sigma_1 = 8$, $\sigma_2 = 4$ and $\sigma_3 = 2$
  - Configuration 2: $\sigma_1 = 10$, $\sigma_2 = 5$ and $\sigma_3 = 2$
  - Configuration 3: $\sigma_1 = 12$, $\sigma_2 = 6$ and $\sigma_3 = 2$
- The start temperature control parameter: $\rho \in \{0.05, 0.1, 0.2\}$
- The reduction factor (cooling rate factor): $r \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$.

Therefore, each of the 20 random instances of N20 was solved by the ALNS with each of the possible $2 \cdot 3^4 \cdot 4 \cdot 5 = 3240$ combinations of the parameters, having in total $3240 \cdot 20 = 64800$ executions of the ALNS for the (MC) problem. After analyzing this experiment, a similar one was done for N80. Due to the size of the networks of N80 and some conclusions obtained from the first experiment, some different values for the parameters of ALNS were tested for the second experiment.

- Number of edges to delete to form neighbors: $n_{del} \in \{1, 2\}$
- The stopping criteria: $n_{iter\_max} = 5$
- Length of a segment: $s \in \{2, 3, 5\}$
- The reaction factor: $\kappa = 0.25$
- Weight adjustment: $(\sigma_1, \sigma_2, \sigma_3) = (8, 4, 2)$
- The start temperature control parameter: $\rho \in \{0.05, 0.1, 0.2\}$
- The reduction factor (cooling rate factor): $r = 0.997$.

For N80, each of the 20 random instances was solved for the $2 \cdot 3^2 = 18$ possible combinations of the parameters. The computational results of this experiment are added as supplementary material in `https://github.com/Natividad13/ALNS_for_Network_Design`.

For each set of instances, an ANOVA analysis is performed to measure whether there is significant evidence between the different levels of the seven parameters considered for each of the variables. Note that for N80, n_{iter_max}, κ, $(\sigma_1, \sigma_2, \sigma_3)$ and r are not factors for the ANOVA analysis since they do not get different levels. The individual p-values obtained are shown in Tables 4.7 and 4.8. For N20, it is observed that there exists significant evidence to confirm that the different levels of n_{del}, n_{iter_max}, s and ρ affect variables `Value` and `RPD`. For N80, there is no significant evidence to accept that neither of the factors considered affect `Value` nor `RPD`, as shown in Table 4.8. With respect to `t`, for N20, all of the factors, except r and κ, are significant for it. However, for N80, the only significant factor is ρ.

	`Value`	`t`	`RPD`
n_{del}	4.96e-12	<2e-16	<2e-16
n_{iter_max}	<2e-16	<2e-16	<2e-16
s	0.0042	<2e-16	0.0426
κ	0.5153	0.1008	0.2805
$(\sigma_1, \sigma_2, \sigma_3)$	0.6245	0.0014	0.2596
ρ	<2e-16	<2e-16	4.57e-12
r	0.9057	0.4345	0.9252

Table 4.7: P-values of the ANOVA for the ALNS of (MC) using N20 instances.

	`Value`	`t`	`RPD`
n_{del}	0.342	0.718	0.999
s	0.332	0.157	0.969
ρ	0.767	7.03e-09	0.914

Table 4.8: P-values of the ANOVA for the ALNS of (MC) using N80 instances.

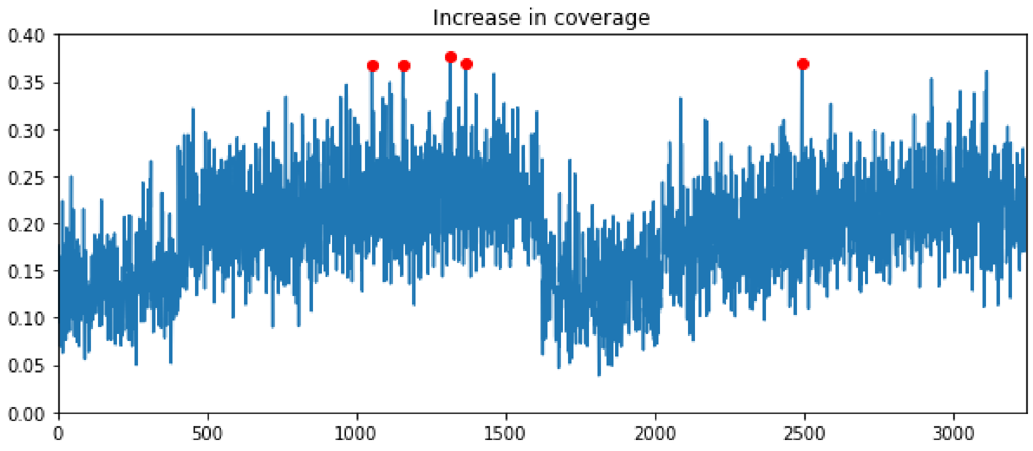


Figure 4.8: Mean values of the overall percentage increase in coverage for all the 3240 combinations of parameters of ALNS over the GRASP for (MC) using N20.

Increase in coverage Figures 4.8 and 4.9 show the mean values of variable `Value` for N20 and N80 considering all the possible combinations of the parameters in each case. That is, for N20, each point in this line graph has an associated 7-tuple $(n_{del}, n_{iter\_max}, \kappa, s, (\sigma_1, \sigma_2, \sigma_3), \rho, r)$. For N80, each point has an associate 3-tuple

(n_{del}, s, ρ). We see that for both sets of instances, all the configurations improve the solution obtained from the GRASP. Intuitively, for N20 the best configurations of these parameters could be $(1, 20, 0.5, 5, 2, 0.05, 0.6)$, $(1, 20, 0.75, 3, 3, 0.05, 0.6)$, $(1, 40, 0.25, 5, 1, 0.1, 0.8)$, $(1, 40, 0.5, 2, 2, 0.05, 0.5)$ and
$(2, 20, 0.25, 3, 2, 0.05, 0.6)$ with an improvement of 36.83%, 36.72%, 37.69%, 36.94%, and 37%, respectively. For N80, the best configuration is $(2, 2, 0.05)$, with an improvement of 6.76%.

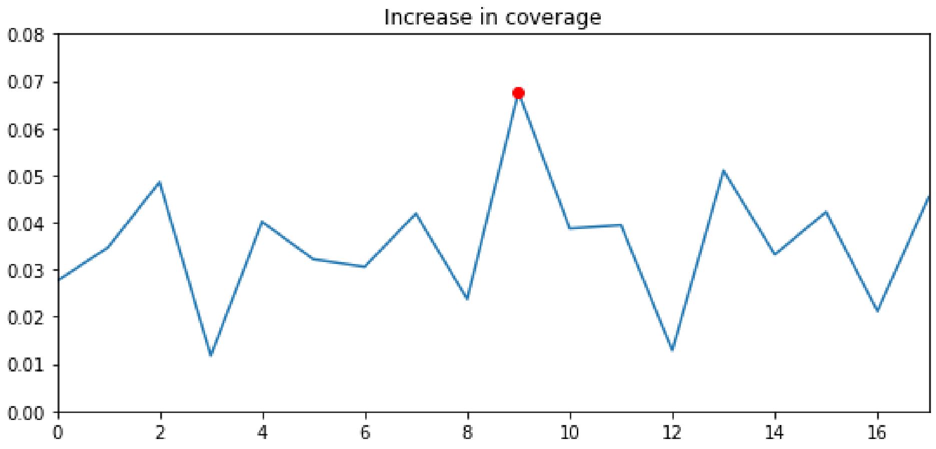


Figure 4.9: Mean values of the overall percentage increase in coverage for all the 216 combinations of parameters of ALNS over the GRASP for (MC) using N80.

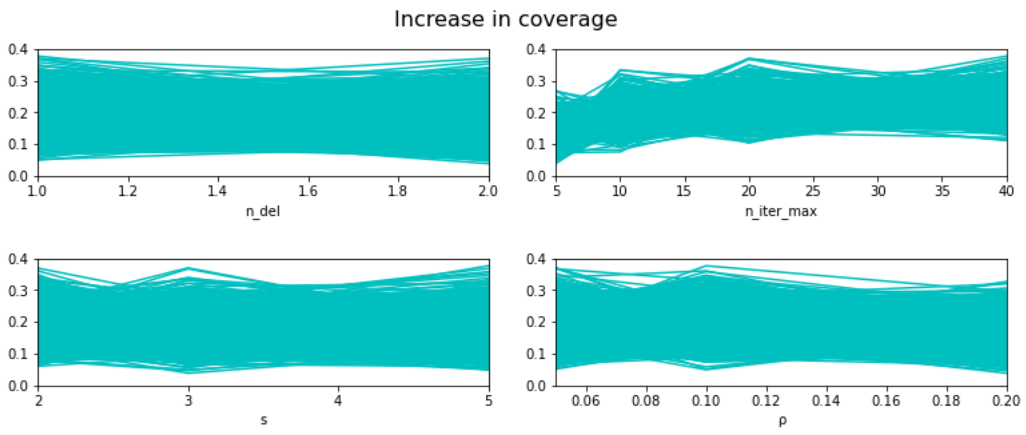


Figure 4.10: Mean values of the overall percentage increase in coverage according to the variation of number of edges removed, number of iterations, length of the segment and reduction factor of ALNS over the GRASP for (MC) using N20.

With respect to obtaining some conclusions about the effect of the different levels of the parameters on the increase in coverage, some observations can be stated from Figure 4.10. Factors analyzed and drawn are the only ones that provide evidence of affect in the previous ANOVAs executed. For N20, we observe that the larger

n_{iter_max} or the shorter ρ, the larger the increase in coverage. Besides, we observe a slightly larger increase for $s = 5$ and $n_{del} = 1$ for the majority of the cases. Since we observe that n_{iter_max} positively influences the larger its value, we fixed it for the experiment with N80. Parameters κ, $(\sigma_1, \sigma_2, \sigma_3)$ and r were also fixed due to their large p-values. In this way, we restricted the analysis to the rest of the parameters, for which there was not enough accentuated trend. Anyway, going back to the ANOVA analysis, none of the factors considered significantly affect `Value`.

Quality of solutions For N80, none of the instances was solved to optimality using the `BD_CW` routine. Observing the collected data for this study, if we compare the best solution found in 1 hour by the `BD_CW` routine with the solutions found by the different configurations of ALNS (whose overall time is less than half an hour), it turns out that, for 53.70% of the cases, the metaheuristic achieves better results (considering the set of all instances used).

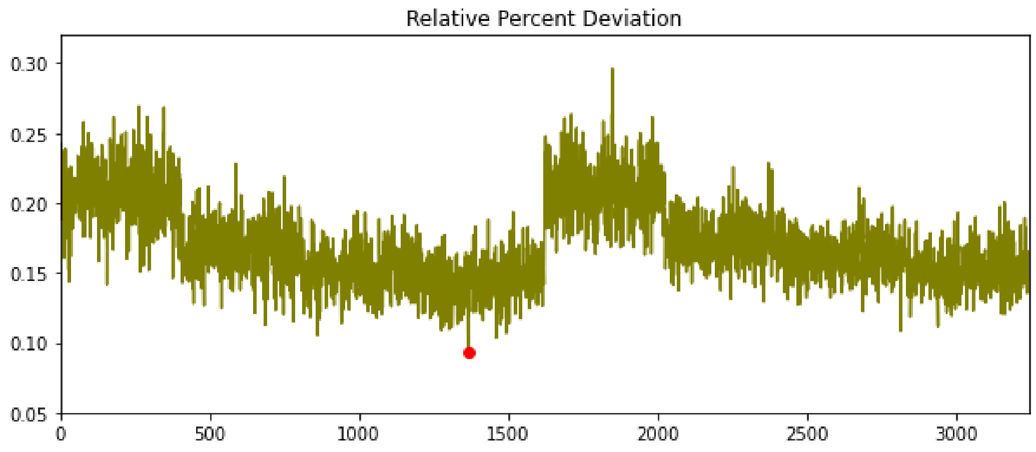


Figure 4.11: Mean values of the overall `RPD` for all the 3240 combinations of parameters of ALNS for (MC) with N20.

In terms of deviation with respect to the solution obtained by the `BD_CW` routine, the ALNS with configuration $(1, 40, 0.5, 1, 2, 0.05, 0.5)$ yields the best results for N20, as shown in Figure 4.11. This configuration finds solutions 9.33% far from the optimal solutions. The previous ANOVA described shows that there is significant evidence to accept that the different values of more than half of the factors affect `RPD`. Nevertheless, for N80, there is no significant evidence to accept that the different values considered for the parameters affect `RPD`. Besides, we observe in Figure 4.12 that the solutions found by ALNS are at least 61.08% far from the best solution found by the `BD_CW` routine after 1 hour. Working with the collected data, we get that this situation is positive for 53.70% of the cases of N80, since for them the solution found by ALNS is better than that reached by using the `BD_CW` routine during 1 hour. This situation is negative for 46.30% of the cases. Thus, we consider two sets. On the one hand, in cases in which ALNS gets better results, being configuration $(1, 5, 0.05)$ the best one. On the other hand, cases for which ALNS has got a solution worse than that achieved by the `BD_CW` routine. For this situation, $(2, 5, 0.1)$ is the best configuration.

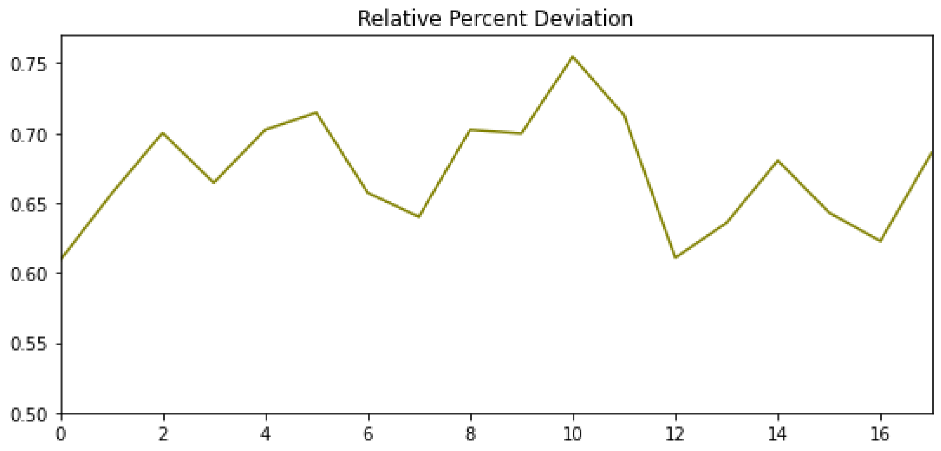


Figure 4.12: Mean values of the overall `RPD` for all the 18 combinations of parameters of ALNS for (MC) with N80.

CPU time Finally, we analyze the computational effort of the ALNS algorithm presented. The average CPU time for all of the combinations tested for the ALNS algorithm are collected in the supplementary material `https://github.com/Natividad13/ALNS_for_Network_Design`. Combinations with a small value of n_{iter_max} and a large value of ρ are those which present a significative short computational time compared with the rest of them. For N20, the average CPU time of the `BD_CW` routine is $53.04s$. Roughly speaking, the ALNS algorithm gets solutions which are on average 9.33% (taking into account the best configuration) far from the MILP (Figure 4.11), in a seventh of the `BD_CW` routine time. The previous ANOVA described shows that more than half of the factors significantly affect the CPU time variable `t`. For N80, the `BD_CW` routine was executing during 1 hour. Taking into account all the combinations tested for ALNS, the average CPU time is $1694s$. Thus, in less than half an hour, for 53.70% of the cases, the ALNS algorithm gets solutions that are on average 67.18% better than the solution obtained by the `BD_CW` routine. However, in half an hour, for 46.30% of the cases, the ALNS algorithm gets solutions which are on average 67.18% worse than that of the `BD_CW` routine. Observing the ANOVA analysis, only the values considered for the factor ρ affect significantly variable `t` using N80 instances.

Summary of results Table 4.9 shows, for the variables studied, the best combinations of ALNS algorithm factors in N20 and N80 instances. As we noted before, not all the factors are individually significant for each variable studied.

Variable	Best combinations for N20	Best combinations for N80
`Value`	$(1, 20, 0.5, 5, 2, 0.05, 0.6)$, $(1, 20, 0.75, 3, 3, 0.05, 0.6)$, $(1, 40, 0.25, 5, 1, 0.1, 0.8)$, $(1, 40, 0.5, 2, 2, 0.05, 0.5)$, $(2, 20, 0.25, 3, 2, 0.05, 0.6)$	$(2, 2, 0.05)$
`t`	All the combinations with $n_{iter\_max} = 5$ and $\rho = 0.2$	All the combinations with $n_{iter\_max} = 5$ and $\rho = 0.2$
`RPD`	$(1, 40, 0.5, 2, 2, 0.05, 0.5)$	$(1, 5, 0.05)$, if $Z_{ALNS} \geq Z_{\texttt{BD\_CW}}$, $(2, 5, 0.1)$, if $Z_{ALNS} < Z_{\texttt{BD\_CW}}$

Table 4.9: Best combinations of ALNS algorithm factors.

Just as we mentioned before, having a large increase in coverage may be due to the fact that the GRASP solution from which ALNS starts is not good. On the other hand, having a `RPD` close to 0 means that the solution provided by the metaheuristic routine is quite good. This may be due to the fact that the GRASP solution from which ALNS starts is also quite good and, in this case, the increase in coverage with respect to the GRASP solution is small. This is exactly what happens for some configurations using N20 instances (see Table 4.10). We determine how the best quality solutions are obtained when a large number of iterations and short values of ρ and s are considered. The good behaviour when considering a short value for ρ could be justified as there are more chances of generating a good initial solution with GRASP. Besides, considering $n_{del} = 1$ seems to be slightly better than setting $n_{del} = 2$. This may be because when setting $n_{del} = 2$, too much variability could be occurring in a small network. Some different conclusions have been obtained for N80, having the best combinations of `RPD` dominance over the rest of them. Note that neither of the factors is significant for `Value` or `RPD`. Nevertheless, for these big instances, as well as for N20, it seems to be better to consider small values of ρ. Besides, large values of s go to better solutions. This difference compared to N20 could be justified by the size of the networks. For small networks, considering short values of s could generate networks very similar to the initial one. With respect to n_{del} there are no clear conclusions nor trends about it. These featured combinations are the ones that will be tested later in Subsection 4.6.6 with a large instance.

Variable	Combination	Value	RPD
	$(1, 20, 0.5, 5, 2, 0.05, 0.6)$	36.83%	12.75%
	$(1, 20, 0.75, 3, 3, 0.05, 0.6)$	36.72%	15.91%
Value	$(1, 40, 0.25, 5, 1, 0.1, 0.8)$	37.69%	12.72%
	$(1, 40, 0.5, 2, 2, 0.05, 0.5)$	36.94%	9.33%
	$(2, 20, 0.25, 3, 2, 0.05, 0.6)$	37%	15.61%
t	All the combinations with $n_{iter\_max} = 5$ and $\rho = 0.2$	around 12.76%	around 20.24%
RPD	$(1, 40, 0.5, 2, 2, 0.05, 0.5)$	36.94%	9.33%

Table 4.10: Analysis of the best combinations of ALNS algorithm factors using N20.

4.6.5 Preliminary experiments for GA

In the experiments of the GA exposed in Perea et al. (2020), the neighborhood structure of the GRASP is tested in three different ways: allow only connected graphs, allow only non-connected graphs and random generation of graphs. In the experiments described in this chapter, the neighborhood structure corresponds to the last form of those named above, i.e., it is composed of connected and non-connected graphs, as explained in Subsection 4.2.

To analyse the performance of GA, we have elaborated a sensitivity analysis for its parameters using instances N20 and N80. We fixed $\alpha = 0.4$. For N20, all the possible combinations of the following chosen values for each of the parameters are tested.

- The stopping criteria: $n_{iter\_max} \in \{20, 40, 80, 160\}$
- Size of the population: $|S| = \{5, 10, 20, 40\}$
- Number of crossover points: $n_{cross} = \{0, 1, 2\}$
- $p_m \in \{0, 0.25, 0.5, 0.75\}$

Therefore, each of the 20 random instances of N20 was solved by the GA with all the possible $4^3 \cdot 3 - 4^2 = 176$ combinations of the parameters, regardless of the case in which the crossover and mutation operators are not executed. Then, there are $176 \cdot 20 = 3520$ executions in total. After observing the results, a similar sensitivity analysis was done for N80. Due to the size of the networks N80 and to some conclusions of the first experiment, some different values for the parameters of GA were tested for the second one.

- The stopping criteria: $n_{iter\_max} = 5$
- Size of the population: $|S| = 5$

- Number of crossover points: $n_{cross} = \{0, 1, 2\}$
- $p_m \in \{0, 0.25, 0.5, 0.75\}$

For N80, each of the 20 random instances was solved with each of the possible $4 \cdot 3 - 1 = 11$ combinations of the parameters, having in total 220 executions of the GA for the (MC) problem. For N80, for bigger values of n_{iter_max} or $|S|$ than those considered, the execution time is larger than 1 hour for some of the instances. Thus, for the analysis, we have considered only the combinations of the parameters in which the computation time is smaller than or equal to 1 hour for all the instances of N80. The computational results of this experiment are added as supplementary material in `https://github.com/Natividad13/GA_for_Network_Design`.

	`Value`	`t`	`RPD`	`Population_Value`
n_{iter_max}	<2e-16	<2e-16	<2e-16	0.689
$\|S\|$	<2e-16	<2e-16	<2e-16	0.152
n_{cross}	<2e-16	<2e-16	<2e-16	0.991
p_m	0.000164	0.165	0.000621	0.899

Table 4.11: P-values of the ANOVA for the GA of (MC) using N20 instances.

	`Value`	`t`	`RPD`	`Population_Value`
n_{cross}	0.0015	0.542	0.999	0.659
p_m	0.1611	0.978	0.999	0.391

Table 4.12: P-values of the ANOVA for the GA of (MC) using N80 instances.

For each set of instances, an ANOVA analysis is performed to measure whether there is significant evidence between the different levels of the number of iterations n_{iter_max}, size of the population $|S|$, number of crossover points n_{cross} and probability of mutation p_m for each of the variables. Note that for N80, n_{iter_max} and $|S|$ are not factors for the ANOVA analysis since they do not get different levels. The individual p-values obtained are shown in Tables 4.11 and 4.12. For N20, it is observed that there exists significant evidence to confirm that the different levels of these four parameters affect variables `Value` and `RPD`. With respect to `t`, the different levels of factors n_{iter_max}, $|S|$ and n_{cross} affect it using N20. For N80, none of the factors considered affect it significantly. For N80, there is significant evidence to accept that factor n_{cross} affects `Value`. However, there is no significant evidence to accept that the different levels of p_m affect `Value`. Besides, none of the factors affect `RPD` nor `t`. Finally, there is no significant evidence to confirm that any of the factors affect `Population_Value` for either of the two sets of instances.

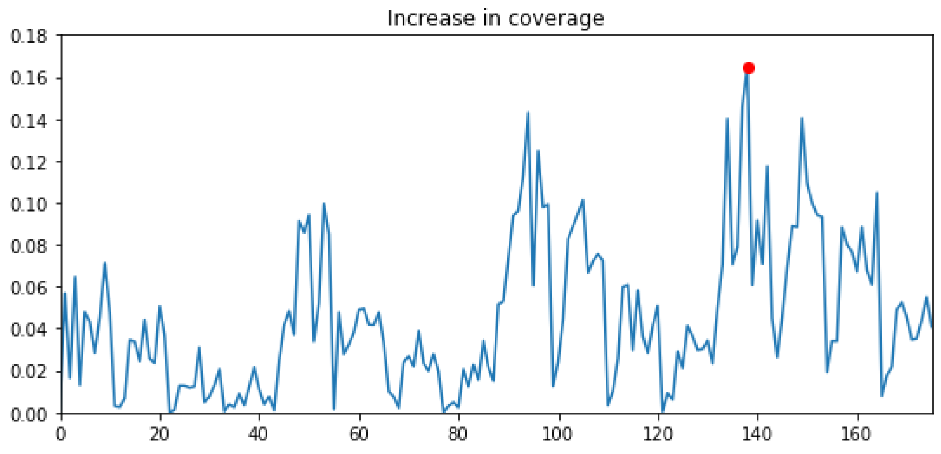


Figure 4.13: Mean values of the overall percentage increase in coverage for all the 176 combinations of parameters of GA over the GRASP for (MC) using N20.

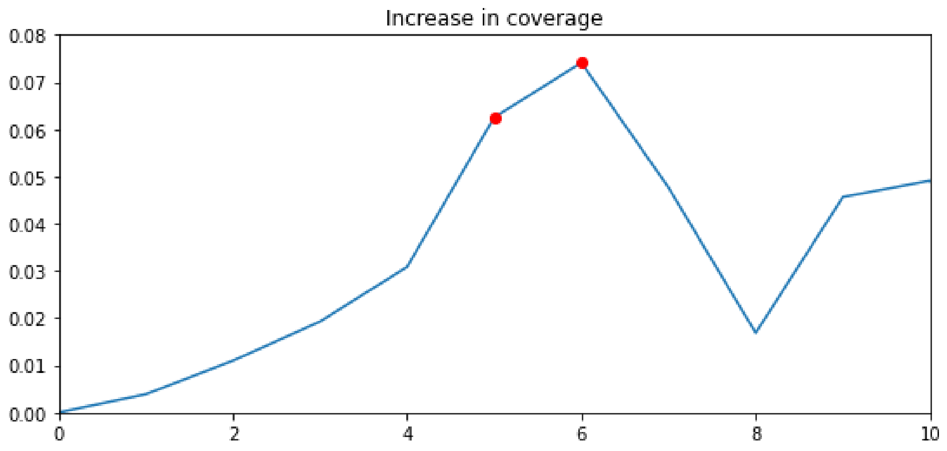


Figure 4.14: Mean values of the overall percentage increase in coverage for all the 11 combinations of parameters of GA over the GRASP for (MC) using N80.

Increase in coverage Figures 4.13 and 4.14 show the mean values of `Value` for N20 and N80 considering all the possible combinations of the parameters in each case. That is, for N20, each point of this line graph has an associated 4-tuple $(n_{iter\_max}, |S|, n_{cross}, p_m)$. For N80, each point has an associated 2-tuple (n_{cross}, p_m), since $n_{iter\_max} = 5$ and $|S| = 5$. We can see that for both sets of instances, almost all of the configurations improve the solution obtained from the GRASP. Intuitively, for N20, the best configuration of these parameters could be $(40, 5, 1, 0.75)$ with an improvement of 16.48%. For N80, the best configurations are $(1, 0.5)$ and $(1, 0.75)$, with an improvement of 6.25% and 7.41%, respectively.

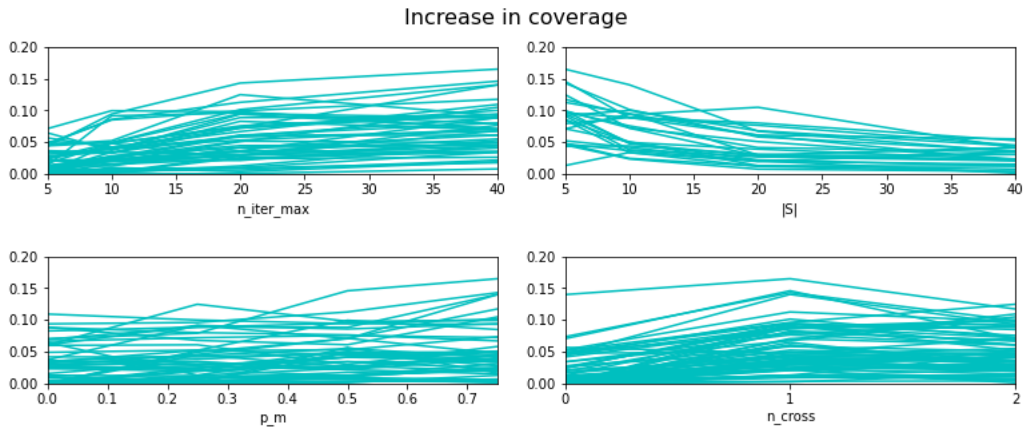


Figure 4.15: Mean values of the overall percentage increase in coverage according to the variation of number of iterations, size of the population and crossover and mutation operators of GA over the GRASP for (MC) using N20.

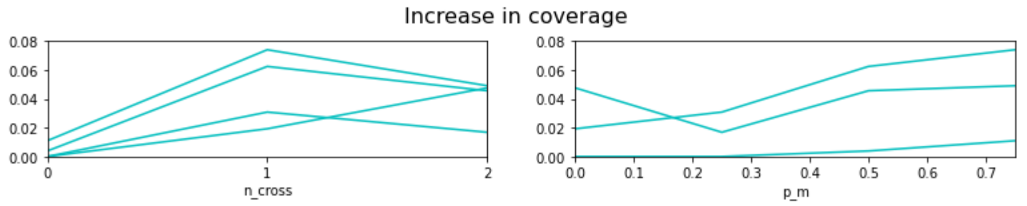


Figure 4.16: Mean values of the overall percentage increase in coverage according to the variation of crossover and mutation operators of GA over the GRASP for (MC) using N80.

By observing Figures 4.15 and 4.16, we obtain some conclusions about the effect of the different levels of the parameters on the increase in coverage. From Figure 4.15, using N20 instances, we observe that fixing $|S|$, p_m and n_{cross}, the larger n_{iter_max}, the larger the increase in coverage, in general. Besides, it is observed clearly that the larger $|S|$, the less the increase in coverage. Moreover, we observe larger values for the increase in coverage for $p_m \geq 0.5$ and $n_{cross} \neq 0$, for the majority of the cases. Going back to the best configuration highlighted in Figure 4.13, note that it exactly corresponds with these previous conclusions. Using N80, it is observed a major improvement if $n_{cross} = 1$ for the majority of the cases. Moreover, it seems that if $p_m \in \{0.5, 0.75\}$ the increase in coverage is larger compared with the situation in which this operator is not applied or has a small p_m assigned, even though the previous ANOVA analysis has not determined significant evidence to accept that the different levels considered of p_m affect `Value`. Looking at the data, we see that the combination which gets null improvement in coverage is characterized by not performing the crossover operator ($n_{cross} = 0$) and $p_m = 0.25$.

Quality of solutions For N80, none of the instances was solved to optimality using the BD_CW routine. Observing the collected data for this study, if we compare the best solution found in 1 hour by the BD_CW routine with the solutions found by the different configurations of GA (whose time is less than 1 hour), it turns out that, for 55.55% of the cases, the metaheuristic achieves better results (considering the set of all instances used). Remember that similar results are obtained for SA, with 59.6% of the cases in favor of it.

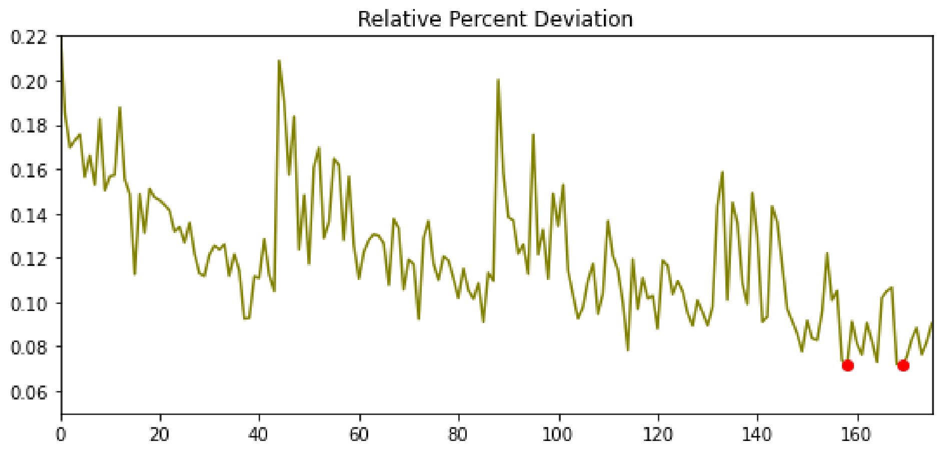


Figure 4.17: Mean values of the overall RPD for all the 176 combinations of parameters of GA for (MC) using N20.

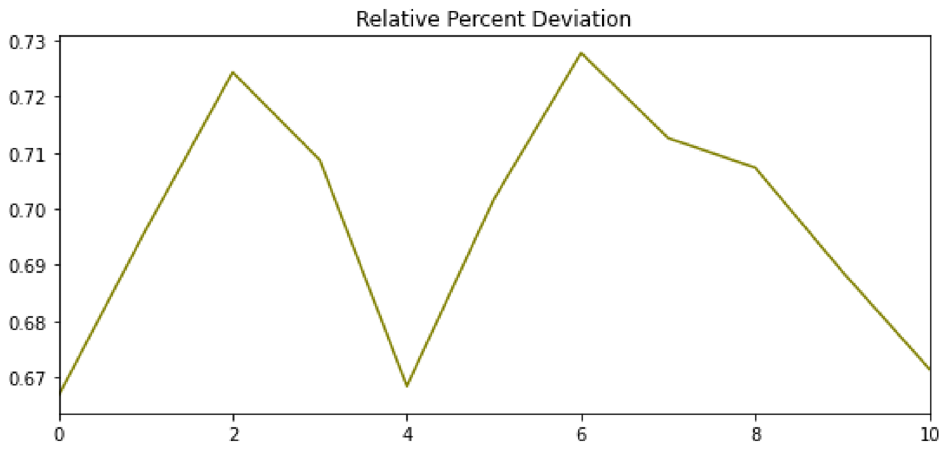


Figure 4.18: Mean values of the overall RPD for all the 22 combinations of parameters of GA for (MC) using N80.

In terms of deviation with respect to the solution obtained by the BD_CW routine, the GA with configurations $(40, 20, 1, 0.25)$ and $(40, 40, 1, 0.25)$ yields the best results using N20, as shown in Figure 4.17. These configurations find solutions only around 7.14% different from the optimal solutions. The previous ANOVA described shows that there is significant evidence to accept that the different values of all the factors affect `RPD`. Nevertheless, for N80, there is no significant evidence to accept that the different values considered for none of the factors affect `RPD`. Besides, we observe in Figure 4.18 that at least the solutions found by GA are 66.66% different from the best solution found by the `BD_CW` routine after 1 hour. Working with the collected data, we get that this situation is positive for 55.55% of the cases of N80, since for them the solution found by GA is better than that reached by using the `BD_CW` routine during 1 hour. This situation is negative for 44.45% of the cases. Thus, we consider two sets. On the one hand, cases in which GA gets better results, being configuration $(2, 0)$ the best one. On the other hand, cases for which GA has got a solution worse than that achieved by the `BD_CW` routine. For this situation, $(2, 0)$ and $(2, 0.75)$ are the best configurations.

CPU time Finally, we analyze the computational effort of the GA presented. The average CPU time for all the combinations tested for the GA are shown in the collected data. Combinations with a small value of $|S|$ are those which present a significative short computational time compared with the rest of them. Fixing $|S|$, there is no significant difference in the computational time. For N20, the average CPU time of the `BD_CW` routine is $53.04s$. Roughly speaking, the GA gets solutions which are on average 7.14% (taking into account the best configuration) far from the MILP solution (Figure 4.17), in a quarter of the `BD_CW` routine time. The previous ANOVA described shows that n_{iter_max}, $|S|$ and n_{cross} significantly affect variable `t`. For the set N80, the `BD_CW` routine was executing during 1 hour. Taking into account all the combinations tested for GA, the average CPU time is $2219s$. Thus, in a little more than half an hour, for 55.55% of the cases, the GA gets solutions which are on average 70.25% better than the solution obtained by the `BD_CW` routine. However, in a little more than half an hour, for 44.45% of the cases, the GA gets solutions which are on average 70.25% worse than that of the `BD_CW` routine.

Summary of results Table 4.13 shows, for the variables studied, the best combinations of GA factors using N20 and N80 instances. As we noted before, not all the factors are individually significant for each variable studied.

Just as we mentioned before, having a large increase in coverage may be due to the fact that the GRASP solution from which GA starts is not good. On the other hand, having a `RPD` close to 0 means that the solution provided by the metaheuristic routine is quite good. This may be due to the fact that the GRASP solution from which GA starts is also quite good and, in this case, the increase in coverage with respect to the GRASP solution is small. This is exactly what happens using N20 if we observe Table 4.14. We observe how the best quality solutions are obtained when a large number of iterations and population are considered, n_{cross} fixed to 1 and $p_m \in \{0.25, 0.75\}$. The good behaviour considering large value for $|S|$ could be justified as there are more chances of generating good initial solutions with GRASP. Different conclusions are obtained for N80. For these instances it seems to be better considering $n_{cross} = 2$. Nevertheless, there are no clear conclusions about the probability of the mutation operator to fix, so the logical recommendation is to always try the three levels of it. These featured combinations are the ones that will be tested later in Subsection 4.6.6 with a large instance.

Variable	Best combinations for N20	Best combinations for N80
`Value`	$(40, 5, 1, 0.75)$	$(1, 0.5)$, $(1, 0.75)$
`t`	All the combinations with $\lvert S\rvert = 5$	All the combinations with $\lvert S\rvert = 5$
`RPD`	$(40, 20, 1, 0.25)$, $(40, 40, 1, 0.25)$	$(2, 0)$ if $Z_{GA} \geq Z_{\texttt{BD\_CW}}$, $(2, 0)$, $(2, 0.75)$ if $Z_{GA} < Z_{\texttt{BD\_CW}}$

Table 4.13: Best combinations of GA factors.

Variable	Combination	`Value`	`RPD`
`Value`	$(40, 5, 1, 0.75)$	16.48%	9.91%
`t`	All the combinations with $\lvert S\rvert = 5$	around 7.26%	around 14.78%
`RPD`	$(40, 20, 1, 0.25)$	8%	7.14%
	$(40, 40, 1, 0.25)$	5.24%	7.14%

Table 4.14: Analysis of the best combinations of GA factors using N20.

4.6.6 Comparative of the metaheuristics on benchmark instances

In this subsection, we have the purpose of comparing the performance of the three metaheuristics analyzed. For that, we use an instance with 100 nodes and 128 edges randomly generated as those described in Subsection 4.6.1. We have elaborated a computational experience according to the change in the value of α.

Firstly, we study the performance of the `BD_CW` approach in Table 4.15, where `v(BD_CW)` represents the best value found by the `BD_CW` procedure in 1 hour. We observe that small values of α make it more difficult to get the optimal solution, or at least a solution, since for $\alpha \in \{0.2, 0.4\}$ none were found. We tried to improve these two situations by providing the `BD_CW` procedure with an initial solution generated with the GRASP, situation that is reflected in the table. We observe that `gap` is very large. Let us remember that `gap` refers to the size of the best bound as a function of the objective value of the best solution found (in 1 hour, in this case). For $\alpha = 0.6$, the instance was solved to optimality.

α	`t`	`gap`	`v(BD_CW)`	`cuts`
0.2	3600	700449.48	133	21752
0.4	3600	667690.43	212	67412
0.6	448.76	0.001	1472095	18736

Table 4.15: Computational performance of `BD_CW` with a time limit of 1 hour using an instance with 100 nodes and 128 edges.

α	Algorithm	Configuration	`Value`
	SA	(1,40,200,0.997)	301165
0.2	ALNS	(2,1,0.1,0.5,0.997)	275817
	GA	(3,0.5)	366822
	SA	(1,40,200,0.997)	830083
0.4	ALNS	(2,1,0.1,0.5,0.997)	772061
	GA	(1,0.5)	725811

Table 4.16: Comparative of the best combinations of SA, ALNS and GA algorithms for (MC) with a randomly generated instance with 100 nodes and 181 edges.

Secondly, the best configurations of the three metaheuristics have been tested. The stopping criteria is the same for all of them. Each metaheuristic has been computing during 1 hour. Each configuration of the selected ones has been computed 10 times. Table 4.16 shows average results for some selected ones. Note that in all the cases presented, the solution obtained by the metaheuristic is better than the one reached with `BD_CW`. Nevertheless, for larger values of α, the `BD_CW` procedure gets the optimal solution network in less than half an hour, which is not competitive with the use of any of the metaheuristics studied. Furthermore, we observe that for small values of α ($\alpha = 0.2$), the GA gets better results than the other two procedures, being ALNS the worst of them. Nevertheless, for bigger values of α ($\alpha = 0.4$) it seems that SA has the best behavior and GA comes in third place. As the value of α increases, the number of feasible solutions increases. It seems that for a high combinatorial component SA gets better solution networks. For larger values of α than 0.6 it is better to use the Benders decomposition approach.

4.7 Conclusions

In this chapter, two well-known metaheuristic techniques have been adapted to the (MC) problem studied in Chapter 2. One of them is a Simulated Annealing algorithm and the other is an Adaptive Large Neighborhood Search procedure. The main difference between them is that the second one works with a larger neighborhood. An extensive sensitivity analysis has been carried out for both using randomly generated medium size instances. In addition, we have added to this computational study the Genetic Algorithm exposed in Perea et al. (2020). The goal has been to observe the behavior of the parameters and to find the best parameter settings in each case. The ANOVA analysis done for each of the metaheuristic showed that not all the parameters are individually significant to the quality of the solutions obtained, according to the set of values considered for such parameters. As a final task, we have added a computational comparison between them using such best configurations for a large instance.

Chapter 5

On λ-Cent-Dians and Generalized-Center for Network Design

This chapter extends the classical notion of λ-cent-dian and generalized-center from Facility Location Theory to the much more complex Network Design area. For that, we consider the problems of building a network under a budget constraint to serve a set of origin/destination pairs instead of single points. Such problems minimize a convex combination and the difference composed of the center and median functions, respectively.

The λ-cent-dian concept aims at studying the trade-off between efficiency and equity. We investigate the properties of the λ-cent-dian and generalized-center solution networks under the lens of equity, efficiency, and Pareto-optimality with respect to the bicriteria center/median problem. We provide a mathematical formulation for $\lambda \geq 0$. Besides, we discuss the bilevel structure of this problem for $\lambda > 1$. Moreover, we describe a procedure to give a complete parametrization of the Pareto-optimality set based on solving two linear formulations. For that, we introduce the new concept of maximum λ-cent-dian. Furthermore, we evaluate the quality of the different solution concepts using some inequality measures, not only from the point of view of efficiency and equity. Finally, we develop and test an branch-and-Benders-cut procedure and a metaheuristic algorithm.

5.1 Introduction

Center and Median problems in graphs and extensions

Center and *Median* problems in graphs and Euclidean spaces constitutes the core of Location Science in the late 50s and 60s of the past century. Whereas median problems aim at maximizing the efficiency of the system, center ones try to maximize the effectiveness or fairness. Given a set of points (demands, customers), either in a graph or in a continuous space with associated weights, the median problem consists in finding one or several points (facilities), so that the normalized sum of the weighted distances from the demand points to the closest facility is minimized. On the other hand, the center problem consists in finding one or several points so that the largest distance from a demand point to the closest facility is minimized. The median notion fits well with problems in which the goal consists of minimizing the cost or maximizing the profit of the system. The goal of the center problems is to minimize the maximum distance. Thus it fits better to locate emergency facilities where the farthest point must be as close as possible to the facility. However, considered isolated, these objectives do not allow to attack many problems in which a balance between both of them is desirable.

In Halpern (1976), the term λ-cent-dian was first introduced for location problems whose objective is to minimize a linear convex combination of center and median objectives, denoted by F_c and F_m, respectively. In other words, the λ-cent-dian objective is $H_\lambda = \lambda F_c + (1-\lambda)F_m$. Subsequently, in Halpern (1978), the author proved that the λ-cent-dian of a graph lies on a path connecting the center and the median. To overcome the drawback of fixing the value of the parameter, and then solving the λ-cent-dian problem, Halpern provided a procedure to find the λ-cent-dians for all possible combinations. A new algorithm to find all λ-cent-dian points is proposed by Hansen et al. (1991). If $\mathcal{N} = (N, E)$ is a graph, then the algorithm finds all the λ-cent-dian points for all $\lambda \geq 0$ in $O(|N||E|\log(|E||N|))$. Moreover, they also introduced the concept of generalized-center as the minimizer of the difference function between the center and the median. In fact, when $\lambda \to \infty$, the ratio $\frac{H_\lambda(x)}{\lambda}$ tends to the generalized-center. Thus, the generalized-center is in some way an equality measure but it could lead to an uninteresting facility location for all the given points. Subsequently, Ogryczak (1997) realized a drawback of the generalized-center and proposed the Chebyshev λ-cent-dian notion.

The λ-cent-dian objective has also been considered in extensive facility location problems. A facility is called extensive if it is too large regarding its environment to be represented by points (see Mesa and Boffey (1996), Puerto et al. (2018)). The first time that research on λ-cent-dian extensive facilities was published was in López-de-los Mozos and Mesa (1992), in which some properties for the λ-cent-dian path in a tree network were derived. The paper Averbakh and Berman (1999) deals with three problems of path location in trees: minimization of a convex combination of the maximal and average distances, and minimizing one objective subject to an upper bound of the value of the other objective. All the problems are solved in $O(n)$ time by the application of some dynamic programming ideas. For more information on path and tree λ-cent-dian problems we refer to Puerto et al. (2009).

The importance of the λ-cent-dian criterion keeps weight to, in some way, contradicting criteria. Thus, the decision-maker chooses the weight to allocate to the center criterion and to the median criterion. In this chapter, such as in all the previous ones, we consider that the demand is given by a set of origin/destination pairs (called O/D pairs), instead of single demand points, for which the selected network is used for connecting them. In Schmidt and Schöbel (2014), the authors present a problem similar to ours for the particular case $\lambda = 1$. That is, the center problem for Network Design.

Most of the Network Design problems already researched have used the cost or the profit as objective functions. These objective functions are surrogates of that of the classical median problems: the sum of the (weighted) distances to the facility since the cost depends on the distance. However, in some cases, it is important that the origins and destinations be not too far away. For example, commuters in a metropolitan area are reluctant to spend daily a lot of time reaching their working places from their homes. This waste of time is also a proxy for distance. Another example happens in electricity distribution generated by (solar, hydro, or wind) mini-power facilities in rural areas. In these cases, the electricity is provided at low voltage and power losses increase with the distance. Thus, in these contexts, the center objective must be considered in combination with the median one.

Bilevel Optimization

Bilevel Optimization is an area of Mathematical Programming constituted by problems which contain a second embedded (nested) problem. That is, some variables are subject to the solution of another optimization problem. The outer optimization task is the so-called upper-level problem, and the inner optimization task is the so-called lower-level problem. These problems involve two kinds of variables.

The general form of a bilevel optimization problem is

$$\min_{\boldsymbol{x}\in X,\boldsymbol{y}} \; F_1(x,y) \tag{5.1}$$

$$\text{s.t. } F_2(x,y) \geq 0, \tag{5.2}$$

$$y \in S(x), \tag{5.3}$$

being $S(x)$ the set of optimal solutions of the $\boldsymbol{x}$-parameterized problem

$$\min_{\boldsymbol{y}\in Y} \; f_1(x,y) \tag{5.4}$$

$$\text{s.t. } f_2(x,y) \geq 0. \tag{5.5}$$

Formulations (5.1)-(5.3) and (5.4)-(5.5) represent the upper-level and lower-level problems, respectively. Moreover, $\boldsymbol{x} \in \mathbb{R}^{n_1}$ and $\boldsymbol{y} \in \mathbb{R}^{n_2}$ are the upper-level and lower-level variables. The objective functions are given by $F_1, f_1 : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$ and the set of constraints by $F_2 : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{m_1}$ as well as $f_2 : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{m_2}$. Sets X and Y denote the integrality constraints.

In our problem, two tasks can be identified. If $\lambda \in [0,1]$, the objective consists of designing a network that minimizes the λ-cent-dian function, which is composed of a sum of travel times of a set of O/D pairs W. Then, implicitly the shortest path for each O/D pair $w \in W$ is identified in the network designed. As we will see later in the chapter, both these objectives are not contrary for the formulation presented with $\lambda \in [0,1]$. However, if $\lambda > 1$, there is a need to enforce the choice of the shortest path for each $w \in W$ because otherwise we get altered objective values or even disturbed solutions. Thus, for $\lambda > 1$, the proposed λ-cent-dian formulation requires the Bilevel Optimization Theory.

There has been a remarkable growth in the interest of computational Bilevel Optimization over the past decade. Dempe and Zemkoho (2020) and Kleinert et al.

(2021) summarize well the theory and applications to real-world problems.

Structure of this chapter

The structure of the chapter is as follows. In Section 5.2, we extend the notions related to the center and median in Network Design. We prove and show that situations similar to those presented in Ogryczak (1997) occur when trying to solve the much more complex problem of designing a network instead of locating a facility. In Section 5.3, we develop mathematical formulation for $\lambda \geq 0$. Besides, we discuss the bilevel structure of this problem for $\lambda > 1$ and we readapt the formulation for this case. Then, we describe a procedure to give a complete parametrization of the Pareto-optimality set based on solving two linear formulations. In Section 5.4, we study two different approaches to solve the problem for the case $\lambda \in [0, 1]$. On the one hand, we tackle the problem using a Benders decomposition implementation together with the ideas exposed in Conforti and Wolsey (2019). On the other hand, we have designed a GRASP algorithm. We present a computational study in Section 5.5 where we investigate the efficiency of the algorithmic tools developed in this work. Besides, this section includes a numerical illustration to discuss the quality of the different solutions concepts developed using some inequality measures, not only from the point of view of the center and median functions. Finally, our conclusions are presented in Section 5.6.

5.2 Problem definition

According to the elements defined in Section 1.2.1, given a graph $\mathcal{N}$, we are interested in subgraphs $\mathcal{S} = (N_{\mathcal{S}}, E_{\mathcal{S}})$ of it, $\mathcal{S} \subseteq \mathcal{N}$, with $E_{\mathcal{S}} \subseteq E \cap (N_{\mathcal{S}} \times N_{\mathcal{S}})$, that can be constructed respecting a budget constraint. We represent this budget as a fraction α of the total cost of building the potential graph $\mathcal{N}$, noted by C_{total}. In other words, a subgraph $\mathcal{S}$ is feasible if

$$\sum_{i \in N_{\mathcal{S}}} b_i + \sum_{e \in E_{\mathcal{S}}} c_e \leq \alpha \left(\sum_{i \in N} b_i + \sum_{e \in E} c_e \right) = \alpha \, C_{total}. \tag{5.6}$$

We denote by $\boldsymbol{\mathcal{N}}^{\alpha}$ the set of all subgraphs of $\mathcal{N}$ satisfying (5.6).

For a given subgraph $\mathcal{S}$ and an O/D pair $w \in W$, $d_{\mathcal{S}}(w)$ represents the length of the shortest path from w^s to w^t in the subgraph $\mathcal{S}$. If a pair $w \in W$ cannot be connected within $\mathcal{S}$, we assume that $d_{\mathcal{S}}(w) = +\infty$. Note that each pair w has an associated utility u^w which represents the length/distance in the alternative existing network. Taking this into account, each user will travel from its origin to its destination in path of length $\ell_{\mathcal{S}}(w) = \min\{d_{\mathcal{S}}(w), u^w\}$. Then if $d_{\mathcal{S}}(w) \leq u^w$, we say that the O/D pair w is *covered* or *served* by $\mathcal{S}$.

Remark 11. *Note that for problems in this chapter, we are assuming that each O/D pair $w \in W$ is always covered by one of the two existing modes which are competing between them. This is a difference with respect to the covering problems studied in the previous chapter. For those problems, only the demand that is covered by the newly designed network is known. The demand that is not covered by the new network may or may not be covered by the existing network.*

Regarding the notation, the notions and properties that are going to be developed in this chapter are supported by network examples in which the couple labeling the edges represents building cost and length, respectively. Also, each node has a label associated with its building cost.

In order to evaluate a solution subgraph $\mathcal{S}$, we now extend two solution concepts coming from Location Science which are the central topic of this work: the *center* and the *median*.

The *average-weighted-O/D-distance* (*median value* to abbreviate) for a set of O/D pairs W in a subnetwork $\mathcal{S} \subseteq \mathcal{N}$ is given by

$$F_m(\mathcal{S}) = \frac{1}{G_{total}} \sum_{w \in W} g^w \ell_{\mathcal{S}}(w). \tag{5.7}$$

Thus, given a potential network $\mathcal{N}$ and a bound $\alpha \in (0,1)$, a subgraph $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ minimizing $F_m(\,\cdot\,)$ is called an *α-Network-Design-median-graph*, *median* to abbreviate, and denoted by $\mathcal{S}_m$. Then, the problem of minimizing $F_m(\,\cdot\,)$ consists of finding a subnetwork $\mathcal{S}_m = (N_{\mathcal{S}_m}, E_{\mathcal{S}_m}) \in \boldsymbol{\mathcal{N}}^\alpha$ such that for every subnetwork $\mathcal{S} = (N_{\mathcal{S}}, E_{\mathcal{S}}) \in \boldsymbol{\mathcal{N}}^\alpha$,

$$\rho_{\mathcal{N}} \equiv \frac{1}{G_{total}} \sum_{w \in W} g^w \ell_{\mathcal{S}}(w) \leq \frac{1}{G_{total}} \sum_{w \in W} g^w \ell_{\mathcal{S}}(w) \equiv \rho_{\mathcal{S}}.$$

It is the so-called α-ND-median-graph problem (median problem to abbreviate).

The *maximum-O/D-distance* (*center value* to abbreviate) for a set of O/D pairs W in a subnetwork $\mathcal{S}$ is denoted by

$$F_c(S) = \max_{w \in W} \ell_{\mathcal{S}}(w). \tag{5.8}$$

Thus, in a potential network $\mathcal{N}$, a subgraph $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ minimizing $F_c(\,\cdot\,)$ is called an *α-ND-center-graph*, *center* to abbreviate, and denoted by $\mathcal{S}_c$. That is, given a potential network $\mathcal{N} = (N, E)$ and the bound $\alpha \in (0,1)$, the problem of minimizing $F_c(\,\cdot\,)$ consists of finding a subnetwork $\mathcal{S}_c = (N_{\mathcal{S}_c}, E_{\mathcal{S}_c}) \in \boldsymbol{\mathcal{N}}^\alpha$, such that for every subnetwork $\mathcal{S} = (N_{\mathcal{S}}, E_{\mathcal{S}}) \in \boldsymbol{\mathcal{N}}^\alpha$,

$$\sigma_{\mathcal{N}} \equiv \max_{w \in W} \ell_{\mathcal{S}}(w) \leq \max_{w \in W} \ell_{\mathcal{S}}(w) \equiv \sigma_{\mathcal{S}}.$$

It is the so-called α-ND-center-graph problem (center problem to abbreviate).

Related to the center value, it can be considered the *weighted-center value* denoted by

$$F_c^G(\mathcal{S}) = \frac{\max\limits_{w\in W}\{g^w \ell_{\mathcal{S}}(w)\}}{g^{w_0}}, \quad w_0 = \arg\max_{w\in W}\{g^w \ell_{\mathcal{S}}(w)\}.$$

Thus, in the same way, we use the concept and the problem of *α-ND-weighted-center-graph*, *weighted-center* problem to abbreviate, whose optimal solution network is denoted by $\mathcal{S}_c^G$.

Definition 3. *Given two subnetworks $\mathcal{S}_1, \mathcal{S}_2 \subseteq \mathcal{N}$, it is said that:*

- *$\mathcal{S}_1$ is more efficient than $\mathcal{S}_2$ iff $F_m(\mathcal{S}_1) < F_m(\mathcal{S}_2)$.*
- *$\mathcal{S}_1$ is less eccentric than $\mathcal{S}_2$ iff $F_c(\mathcal{S}_1) < F_c(\mathcal{S}_2)$.*
- *$\mathcal{S}_1$ is less weighted-eccentric than $\mathcal{S}_2$ iff $F_c^G(\mathcal{S}_1) < F_c^G(\mathcal{S}_2)$.*

Given that the center minimizes the maximum travel time, it often leads to inefficient solutions. This inefficiency is produced by not considering feasible subgraphs in which travel time decreases for some users maintaining the center objective function value. We depict this situation in Example 1.

Example 1. Given the bound $\alpha\, C_{total} = 90$ and the following potential network, we want to minimize the median and the center values.

Origin	Destination	u^w	g^w
1	6	92	200
2	5	92	50
4	1	92	50

Table 5.1: Data in Example 1. We consider $\alpha\, C_{total} = 90$.

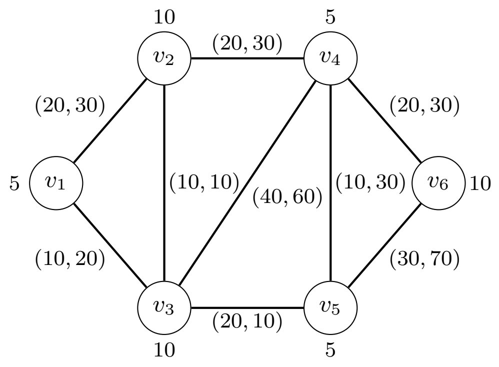


Figure 5.1: Network in Example 1.

We observe that with the given bound on the total cost it is not possible to serve the three O/D pairs at the same time. Hence, the objective value for the center is $F_c(\mathcal{S}) = 92$. Note that the empty subgraph, $\mathcal{S}_0 = \emptyset$, has a center value $F_c(\mathcal{S}_0) = 92$ and it is thus an optimal solution for the center problem. In other words, not building anything can be optimal for the center problem. For this solution, the median objective has also a value of $F_m(\mathcal{S}_0) = 92$. Now let us consider the following two solutions:

- $\mathcal{S}_1$ composed of node set $N_1 = \{v_1, v_2, v_4, v_6\}$ and edge set $E_1 = \{(v_1, v_2), (v_2, v_4), (v_4, v_6)\}$ such that O/D pairs $(1, 6)$ and $(4, 1)$ are served; and
- $\mathcal{S}_2$ with $N_2 = \{v_1, v_2, v_4, v_5\}$ and $E_2 = \{(v_1, v_2), (v_2, v_4), (v_4, v_5)\}$ such that O/D pairs $(2, 5)$ and $(4, 1)$ are served.

Observe that both solutions have a center value of $F_c(\mathcal{S}_1) = F_c(\mathcal{S}_2) = 92$, but the median objective value changes. Indeed, these values are $F_m(\mathcal{S}_1) \approxeq 85.33$ and $F_m(\mathcal{S}_2) \approxeq 81.33$ respectively.

Example 1 shows the necessity of having a finer conceptualization to consider solutions that are not dominated.

Since the median approach is based on averaging, it often provides solutions in which the O/D pairs with too long shortest path and low-demand are discriminated against in terms of accessibility to public networks, as compared with those that have its nodes closer and with high-demand. On the other hand, generating the center could lead to a substantial loss in efficiency. These observations are shown in Example 2.

Example 2 Consider the network and its data summarized in Table 5.2 and Figure 5.2. In this example minimizing the median function does not serve the most distant pairs or those which have a low demand. Besides, minimizing the center function entails an increase in the median value.

Origin	Destination	u^w	g^w
1	6	92	5
2	3	40	65
4	1	50	50

Table 5.2: Data in Example 2. We consider $\alpha\, C_{total} = 90$.

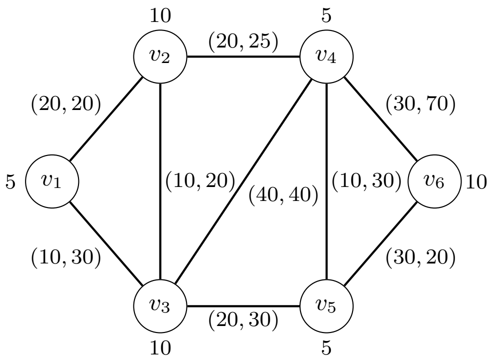
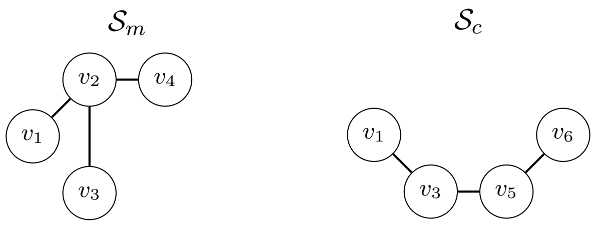


Figure 5.2: Network in Example 2.

We observe that the optimal median and the center, being $\alpha\, C_{total} = 90$, are the ones depicted below.



Median and center values for each subnetwork are:

$$F_m(\mathcal{S}_m) = \frac{1}{120}(5 \cdot 92 + 65 \cdot 20 + 50 \cdot 45) = \frac{4010}{120} \approxeq 38.19, \quad F_c(\mathcal{S}_m) = 92,$$

$$F_m(\mathcal{S}_c) = \frac{1}{120}(5 \cdot 80 + 65 \cdot 40 + 50 \cdot 50) = \frac{5500}{120} \approxeq 52.38, \quad F_c(\mathcal{S}_c) = 80.$$

Even if we consider the weighted case of the center function, the loss in efficiency could be preserved, as in this particular situation. That is, the optimal weighted-center is just $\mathcal{S}_c$ with $F_c^G(\mathcal{S}_c) = 40$.

Example 2 shows that there is a need to capture the trade-off between both solution concepts.

While the median tends to favor users concentrated in the center of the network to the detriment of users in extreme zones, the center favors users in extreme zones regardless of the efficiency of the design. In order to get solutions that balance these two criteria, researchers in Location Science have studied since the 70s the concept of λ-*cent-dian*, that is, a convex combination of the center and median objectives (see Halpern (1976)). Besides, authors in Hansen et al. (1991), introduce the *generalized-center* in the context of Facility Location to reduce as many as possible discrepancies in accessibility among users. Below, the extended concepts α-*ND-generalized-center-graph* and (α, λ)-*ND-cent-dian-graph* are presented.

On the one hand, the *generalized-maximum-O/D-distance* (*generalized-center value* to abbreviate) for a set of O/D pairs W in a subnetwork $\mathcal{S}$ is denoted by

$$F_{gc}(\mathcal{S}) = F_c(\mathcal{S}) - F_m(\mathcal{S}). \tag{5.9}$$

Given a potential network $\mathcal{N}$, a subgraph $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ minimizing $F_{gc}(\,\cdot\,)$ is called an α-*ND-generalized-center-graph*, *generalized-center* to abbreviate. If we use function $F_c^G(\mathcal{S})$ instead of function $F_c(\mathcal{S})$, an α-*ND-generalized-weighted-center-graph* is a subgraph $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ minimizing

$$F_{gc}^G(\mathcal{S}) = |F_c^G(\mathcal{S}) - F_m(\mathcal{S})|. \tag{5.10}$$

In the context of efficiency, the optimal solution to the generalized-center problem could be identified as an unreasonable network design with respect to the distance of the paths (see Example 3 of Subsection 5.2.1). To avoid such solution networks, the optimal solution is restricted to the set of solution networks so-called as *Pareto-optimal with respect to the distances.*

Definition 4. *A subnetwork $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ is Pareto-optimal with respect to the distance of the shortest paths if there does not exist another subnetwork $\mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$ for which*

$$\min\{d_{\mathcal{S}'}(w), u^w\} \leq \min\{d_{\mathcal{S}}(w), u^w\} \text{ for all } w \in W,$$

where at least one of the inequalities is strictly satisfied.

The set of subnetworks $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ that are Pareto-optimal with respect to the distance of the shortest paths is named as $\boldsymbol{\mathcal{PO}}^\alpha$. Thus, the generalized-center of a given network $\mathcal{N}$ is redefined as an optimal solution to the problem

$$\min\{F_c(\mathcal{S}) - F_m(\mathcal{S}) : \mathcal{S} \in \boldsymbol{\mathcal{PO}}^\alpha\}. \tag{5.11}$$

However, Example 4 of the next subsection shows that this reduction in the set of solution networks is not enough to get the most efficient network.

On the other hand, a linear combination of $F_m(\,\cdot\,)$ and $F_c(\,\cdot\,)$ can be taken into account:

$$H_\lambda(\mathcal{S}) = \lambda\, F_c(\mathcal{S}) + (1-\lambda)\, F_m(\mathcal{S}) \quad \text{with} \quad \lambda \geq 0, \tag{5.12}$$

named as the λ-*cent-dian-distance* in $\mathcal{S}$. A subgraph $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ minimizing $H_\lambda(\,\cdot\,)$ is called an (α,λ)-*ND-cent-dian-graph*, λ-*cent-dian* to abbreviate. In this way, for $\lambda \in [0,1]$, a convex combination is minimized. Depending on the weight assigned to each part of the objective function, more importance will be given to minimizing the length of the worst path or to minimizing the average length of the paths. Particularly, if $\lambda = 0$ the λ-cent-dian is a median network and if $\lambda = 1$ it is a center network.

For $\lambda \in (0,1)$, the λ-cent-dian solution concept may be viewed as the weighting approach to the bicriteria *center/median* model

$$\min\{[F_c(\mathcal{S}), F_m(\mathcal{S})] : \mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha\} \tag{5.13}$$

where both, $F_c(\,\cdot\,)$ and $F_m(\,\cdot\,)$, have to be minimized.

Definition 5. *Given a network $\mathcal{N}$, a subnetwork $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ is a Pareto-optimal network with respect to the bicriteria center/median problem if there does not exist another subnetwork $\mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$, for which*

$$F_c(\mathcal{S}') \leq F_c(\mathcal{S}) \text{ and } F_m(\mathcal{S}') \leq F_m(\mathcal{S}),$$

where at least one of the inequalities is strictly satisfied.

The set of subnetworks $\mathcal{S} \subseteq \boldsymbol{\mathcal{N}}^\alpha$ Pareto-optimal with respect to the bicriteria center/median problem is named as $\boldsymbol{\mathcal{PO}}_2^\alpha$. Proposition 5 relates $\boldsymbol{\mathcal{PO}}^\alpha$ and $\boldsymbol{\mathcal{PO}}_2^\alpha$.

For $\lambda > 1$, the solution concept heavily depends on the minimization of the difference $F_c(\mathcal{S}) - F_m(\mathcal{S})$, since $H_\lambda(\mathcal{S})$ can be written as $H_\lambda(\mathcal{S}) = F_m(\mathcal{S}) + \lambda\,(F_c(\mathcal{S}) -$

$F_m(\mathcal{S})$) (see Hansen et al. (1991)).

Proposition 5. *Given a bound α and a potential network $\mathcal{N}$, $\mathcal{PO}_2^\alpha \subseteq \mathcal{PO}^\alpha$.*

Proof. Let us prove by *Reductio ad absurdum.* Let $\mathcal{S} \in \mathcal{PO}_2^\alpha$ and $\mathcal{S} \notin \mathcal{PO}^\alpha$. Then,

- $\nexists \mathcal{S}' \in \mathcal{PO}_2^\alpha$ such that $F_c(\mathcal{S}') \leq F_c(\mathcal{S})$ and $F_m(\mathcal{S}') \leq F_m(\mathcal{S})$, where at least one of the inequalities is satisfied strictly, and
- $\exists\, w' \in W$ and $\exists \mathcal{S}'' \in \mathcal{N}^\alpha$ such that $\ell_{\mathcal{S}''}(w') < \ell_{\mathcal{S}}(w')$ and $\ell_{\mathcal{S}''}(w) \leq \ell_{\mathcal{S}}(w)$, $w \in W \setminus \{w'\}$.

Thus, by construction of functions $F_c(\,\cdot\,)$ and $F_m(\,\cdot\,)$, $F_c(\mathcal{S}'') \leq F_c(\mathcal{S})$ and $F_m(\mathcal{S}'') \leq F_m(\mathcal{S})$. That is, $\mathcal{S} \notin \mathcal{PO}_2^\alpha$, which is a contradiction. □

5.2.1 The generalized-center problem

In this subsection, we analyze the solution concepts of generalized-center and λ-cent-dian associated with $\lambda > 1$, given a general network. In order to avoid "unreasonable" network designs when minimizing these functions (see Example 3), the solution network is restricted to the set $\mathcal{PO}^\alpha$. Anyway, both these solution concepts depend on the minimization of the difference $F_c(\,\cdot\,) - F_m(\,\cdot\,)$ which, in general, does not comply with the bicriteria minimization model (5.13), as shown in Example 4.

Example 3 Let us consider the following network and its associated data described in Table 5.3 and Figure 5.3. It shows that minimizing the generalized-center function carries out unreasonable solutions with respect to the notion of efficiency if the optimal solution is not restricted to the set $\mathcal{PO}^\alpha$.

Origin	Destination	u^w	g^w
1	2	92	50
2	6	100	5
4	1	92	50

Table 5.3: Data in Example 3. We consider $\alpha\, C_{total} = 90$.

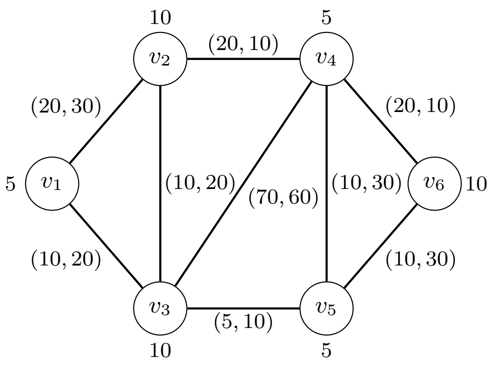
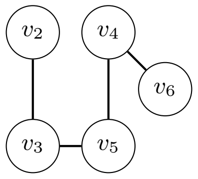


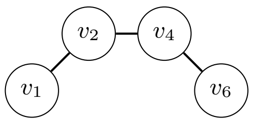
Figure 5.3: Network in Example 3.

The generalized-center, $\mathcal{S}_{gc}$, in this case is:



$F_c(\mathcal{S}_{gc}) - F_m(\mathcal{S}_{gc}) = 92 - \frac{1}{105}(50 \cdot 92 + 5 \cdot 70 + 50 \cdot 92) \approxeq 92 - 90.95.$

Nevertheless, there exists a network, $\mathcal{S}_1$, more efficient (in terms of the distances of the shortest paths) than $\mathcal{S}_{gc}$:



$F_c(\mathcal{S}_1) - F_m(\mathcal{S}_1) = 40 - \frac{1}{105}(50 \cdot 30 + 5 \cdot 20 + 50 \cdot 40) \approxeq 40 - 34.28$

In fact, $\mathcal{S}_1$ is simultaneously the center and the median solution network. Even if we minimize $F_{gc}^G(\,\cdot\,)$, instead of $F_{gc}(\,\cdot\,)$, the optimal solution does not change, which is $\mathcal{S}_{gc}$. Network $\mathcal{S}_1$ is Pareto-optimal with respect to the distances; but $\mathcal{S}_{gc}$ is not. Even more, each of the O/D pairs has its shortest path shorter in $\mathcal{S}_1$ than in $\mathcal{S}_{gc}$. Locating $\mathcal{S}_{gc}$ can worsen the center and median values.

However, even if the problem of the generalized-center is restricted to the set $\mathcal{PO}^\alpha$, very inefficient solution graphs can be obtained as optimal solutions, as we show in Example 4. Having the minimization of the difference $F_c(\,\cdot\,) - F_m(\,\cdot\,)$ and several possible solution networks with the same value $F_c(\,\cdot\,)$, the one with the worst value of $F_m(\,\cdot\,)$ will be selected among them.

Example 4 Given the potential network $\mathcal{N}$ summarized in Table 5.4 and Figure 5.4, in this example we show a situation in which minimizing the generalized-center function over the set $\mathcal{PO}^\alpha$ results in a not very efficient solution.

Origin	Destination	u^w	g^w
1	2	35	50
2	4	35	30
3	1	35	30
4	3	35	20

Table 5.4: Data in Example 4. We consider $\alpha\, C_{total} = 50$.

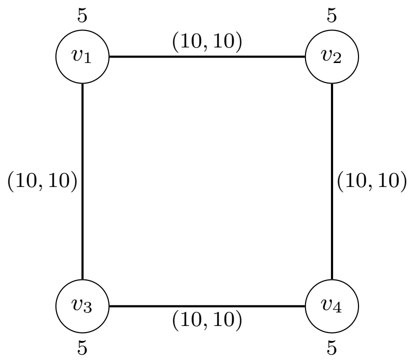
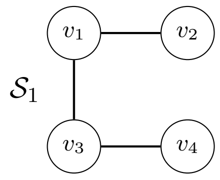
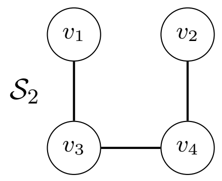


Figure 5.4: Network in Example 4.

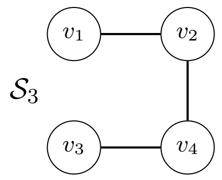
The set $\mathcal{PO}^\alpha$ consists of:



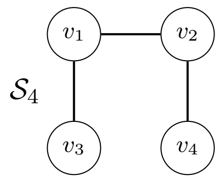
$F_c(\mathcal{S}_1) - F_m(\mathcal{S}_1) = 30 - \frac{1}{130}(50{\cdot}10 + 30{\cdot}30 + 30{\cdot}10 + 20{\cdot}10) = 30 - \frac{1900}{130} \approxeq 15.38$



$F_c(\mathcal{S}_2) - F_m(\mathcal{S}_2) = 30 - \frac{1}{130}(50{\cdot}30 + 30{\cdot}10 + 30{\cdot}10 + 20{\cdot}10) = 30 - \frac{2300}{130} \approxeq 12.30$



$F_c(\mathcal{S}_3) - F_m(\mathcal{S}_3) = 30 - \frac{1}{130}(50{\cdot}10 + 30{\cdot}10 + 30{\cdot}30 + 20{\cdot}10) = 30 - \frac{1900}{130} \approxeq 15.38$



$F_c(\mathcal{S}_4) - F_m(\mathcal{S}_4) = 30 - \frac{1}{130}(50{\cdot}10 + 30{\cdot}10 + 30{\cdot}10 + 20{\cdot}30) = 30 - \frac{1430}{130} \approxeq 19$

These four solution networks are in $\mathcal{PO}^\alpha$ and all of them are centers. Network $\mathcal{S}_2$ is the generalized-center, but $\mathcal{S}_4$ is the most efficient network. One may think the presented flaws in the problem are resolved if we consider the weighted function, but it is not so. The generalized-weighted-center is also $\mathcal{S}_2$:

$$\begin{aligned}
\left|F_c^G(\mathcal{S}_1) - F_m(\mathcal{S}_1)\right| &= \left|30 - \frac{1}{130}(50\cdot 10 + 30\cdot 30 + 30\cdot 10 + 20\cdot 10)\right| = \left|30 - \frac{1900}{130}\right| = 15.38,\\
\left|F_c^G(\mathcal{S}_2) - F_m(\mathcal{S}_2)\right| &= \left|30 - \frac{1}{130}(50\cdot 30 + 30\cdot 10 + 30\cdot 10 + 20\cdot 10)\right| = \left|30 - \frac{2300}{130}\right| \approxeq 12.30,\\
\left|F_c^G(\mathcal{S}_3) - F_m(\mathcal{S}_3)\right| &= \left|30 - \frac{1}{130}(50\cdot 10 + 30\cdot 10 + 30\cdot 30 + 20\cdot 10)\right| = \left|30 - \frac{1900}{130}\right| = 15.38,\\
\left|F_c^G(\mathcal{S}_4) - F_m(\mathcal{S}_4)\right| &= \left|30 - \frac{1}{130}(50\cdot 10 + 30\cdot 10 + 30\cdot 10 + 20\cdot 30)\right| = \left|30 - \frac{1430}{130}\right| \approxeq 19.
\end{aligned}$$

Regarding the efficiency, let us define the *α-ND-restricted-generalized-center-graph* (*restricted-generalized-center* to abbreviate) as an optimal solution to the problem

$$\min\{F_c(\mathcal{S}) - F_m(\mathcal{S}) : \mathcal{S} \in \mathcal{PO}_2^\alpha\}. \tag{5.14}$$

Similarly to the case of the restricted-generalized-center, we introduce the (α,λ)-*ND-restricted-cent-dian-graph* (*λ-restricted-cent-dian* to abbreviate) defined as an optimal solution to the problem

$$\min\{H_\lambda(\mathcal{S}) : \mathcal{S} \in \mathcal{PO}_2^\alpha\}. \tag{5.15}$$

For $\lambda \in (0,1)$, the corresponding λ-cent-dian always belongs to the set $\mathcal{PO}_2^\alpha$. This can be easily proved by *Reductio ad absurdum.* Thus, in this case, the

λ-restricted-cent-dian is simply a λ-cent-dian.

The above is not true for $\lambda \geq 1$, specifically, for the limiting case $\lambda \to \infty$. As shown in Example 4, the generalized-center does not belong to the set $\boldsymbol{\mathcal{PO}}_2^\alpha$ in general. Besides, we will show that for all $\lambda \geq 1$, including the limiting case of the restricted-generalized-center, the corresponding λ-restricted-cent-dian is always a center. Note that, given a bound α, the center of a general network $\mathcal{N}$ may be non-unique and, in such case, not all centers belong to $\boldsymbol{\mathcal{PO}}_2^\alpha$. This issue is shown in Example 4. We know that $\boldsymbol{\mathcal{PO}}^\alpha = \{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3, \mathcal{S}_4\}$, but $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3 \notin \boldsymbol{\mathcal{PO}}_2^\alpha$. To belong to $\boldsymbol{\mathcal{PO}}_2^\alpha$, the center must be unique or it must be the center with the best value of $F_m(\,\cdot\,)$. Thus, the center belonging to $\boldsymbol{\mathcal{PO}}_2^\alpha$ is an optimal solution to the following lexicographic (two-level) problem

$$\operatorname{lex\,min}\{[F_c(\mathcal{S}), F_m(\mathcal{S})] : \mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha\}. \tag{5.16}$$

The lexicographic minimization in (5.16) means that first, we minimize $F_c(\,\cdot\,)$ on the set $\boldsymbol{\mathcal{N}}^\alpha$, and then we minimize $F_m(\,\cdot\,)$ on the optimal subnetworks set of function $F_c(\,\cdot\,)$. The second minimization is only carried out when the optimal solution of $F_c(\,\cdot\,)$ is not unique. We call the optimal solution of (5.16) a *lexicographic cent-dian.* Note that the lexicographic cent-dian is a center and the unique center is the lexicographic cent-dian.

Proposition 6. *Given a bound α, on any network $\mathcal{N}$, the restricted-generalized-center, as well as any λ-restricted-cent-dian for $\lambda \geq 1$, is a lexicographic cent-dian. Conversely, any lexicographic cent-dian is a restricted-generalized-center, as well as a λ-restricted-cent-dian for any $\lambda \geq 1$.*

Proof. Let $\mathcal{S}$ be the lexicographic cent-dian. That is, $F_c(\mathcal{S}) \leq F_c(\mathcal{S}')$ for any $\mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$. Besides, if it turns out that $\exists \mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$ such that $F_c(\mathcal{S}) = F_c(\mathcal{S}')$, then $F_m(\mathcal{S}) \leq F_m(\mathcal{S}')$. Firstly, observe that $\mathcal{S} \in \boldsymbol{\mathcal{PO}}_2^\alpha$. To continue, we consider $\mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$ a solution network of $\boldsymbol{\mathcal{PO}}_2^\alpha$. If $\mathcal{S}'$ is not a lexicographic cent-dian, then $F_c(\mathcal{S}) < F_c(\mathcal{S}')$ or, if $F_c(\mathcal{S}) = F_c(\mathcal{S}')$ then $F_m(\mathcal{S}) < F_m(\mathcal{S}')$. This second situation is not possible since $\mathcal{S}' \in \boldsymbol{\mathcal{PO}}_2^\alpha$. Hence, being in $\boldsymbol{\mathcal{PO}}_2^\alpha$, $\mathcal{S}'$ has to satisfy inequalities

$$F_c(\mathcal{S}) < F_c(\mathcal{S}') \quad \text{and} \quad F_m(\mathcal{S}) > F_m(\mathcal{S}').$$

Thus, $F_c(\mathcal{S}') - F_m(\mathcal{S}') > F_c(\mathcal{S}) - F_m(\mathcal{S})$, which means that the restricted-generalized-center is a lexicographic cent-dian. Moreover, for $\lambda \geq 1$, this property holds:

$$H_\lambda(\mathcal{S}') = F_c(\mathcal{S}') + (\lambda - 1)(F_c(\mathcal{S}') - F_m(\mathcal{S}')) > F_c(\mathcal{S}) + (\lambda - 1)(F_c(\mathcal{S}) - F_m(\mathcal{S})) = H_\lambda(\mathcal{S}),$$

which proves that the corresponding λ-restricted-cent-dian is a lexicographic cent-dian. To finish the proof, since all lexicographic cent-dians have the same center value and also the same median value, they are all restricted-generalized-centers, as well as λ-restricted-cent-dians with $\lambda > 1$. $\square$

Proposition 6 provides us with a very simple characteristic of the λ-restricted-cent-dians for $\lambda \geq 1$. From this, we conclude that the λ-restricted-cent-dians for $\lambda \geq 1$ are simply the centers with the best median value. On the other hand, it means that the solution concept of the restricted-generalized-center does not provide us with any compromise between the median value and center value.

5.2.2 Compromise on λ-cent-dians

In order to provide some compromise between spatial efficiency (median value) and spatial equity (center value) the extended concept λ-*cent-dian* is introduced as the solution network of minimization of (5.12), being $\lambda \in (0, 1)$. In the case of considering a tree network, such compromise is given only for two networks.

In the following example we observe that in a general network there are subnetworks belonging to $\mathcal{PO}_2^\alpha$ different from $\mathcal{S}_c$ and $\mathcal{S}_m$. Besides, this example shows us that it could happen that for some of these subnetworks there does not exist $0 \leq \lambda \leq 1$ such that $\mathcal{S}$ is the corresponding λ-cent-dian.

Example 5 Given the potential network $\mathcal{N}$ and its associated data of Table 5.5 and Figure 5.5, this example shows that minimizing $H_\lambda(\cdot)$ does not give us any compromise between $F_c(\cdot)$ and $F_m(\cdot)$.

Origin	Destination	u^w	g^w
1	2	70	90
3	4	55	20
5	6	92	5

Table 5.5: Data in Example 5. We consider $\alpha\, C_{total} = 70$.

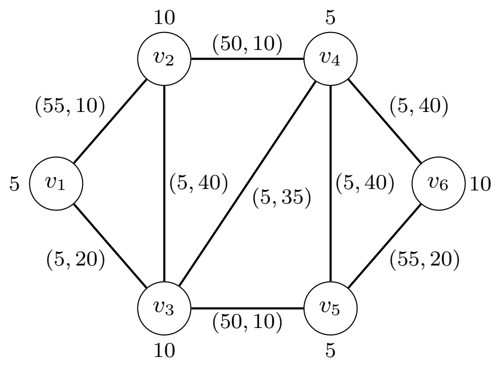
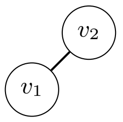


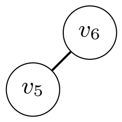
Figure 5.5: Network in Example 5.

The median $\mathcal{S}_m$ is



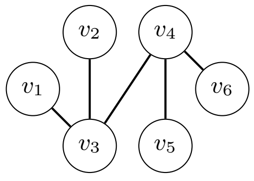
$$F_m(\mathcal{S}_m) = \frac{1}{115}(90 \cdot 10 + 20 \cdot 55 + 5 \cdot 92) = \frac{2460}{115} \approxeq 21.39$$
$$F_c(\mathcal{S}_m) = 92$$

The network has a unique center $\mathcal{S}_c$ with the subnetwork



$$F_m(\mathcal{S}_c) = \tfrac{1}{115}(90 \cdot 70 + 20 \cdot 55 + 5 \cdot 20) = \tfrac{6600}{115} \approxeq 65.21$$
$$F_c(\mathcal{S}_c) = 70$$

There is another subnetwork belonging to $\boldsymbol{\mathcal{PO}}_2^\alpha$. The following network $\mathcal{S}_1$ has a better value for the average of the shortest paths than $\mathcal{S}_c$ and its center value is lower than that of $\mathcal{S}_m$.



$$F_m(\mathcal{S}_1) = \tfrac{1}{115}(90 \cdot 60 + 20 \cdot 35 + 5 \cdot 80) = \tfrac{6425}{115} \approxeq 56.52$$
$$F_c(\mathcal{S}_1) = 80$$

Therefore, for any $0 \leq \lambda \leq 1$, $H_\lambda(\mathcal{S}_m) < H_\lambda(\mathcal{S}_1)$ or $H_\lambda(\mathcal{S}_c) < H_\lambda(\mathcal{S}_1)$. That is, even though $\mathcal{S}_1 \in \boldsymbol{\mathcal{PO}}_2^\alpha$, it cannot be a λ-cent-dian for any $0 \leq \lambda \leq 1$.

Even if we use the weighted function $F_c^G(\cdot)$ the flaw remains. We obtain that the weighted-center coincides with $\mathcal{S}_m$, since $F_c^G(\mathcal{S}_m) = 55$, $F_c^G(\mathcal{S}_c) = 70$ and $F_c^G(\mathcal{S}_1) = 60$. The same solution is obtained whatever the value of the parameter is in the interval $[0, 1]$. It does not result in any compromise between the value of the parameter λ and the functions used. That is, the solution does not change according to the different values that the parameter λ takes.

The situation exposed in Example 5 is not possible if we are considering a tree network, as proven in Proposition 7. Without loss of generality, in Proposition 7 we mean by $\mathcal{S}_c$ (resp. $\mathcal{S}_m$) the unique center (resp. median) or the center (resp. median) with the best value of $F_m(\cdot)$ (resp. $F_c(\cdot)$).

Proposition 7. *Given a bound α, on a tree network $\mathcal{T}$, the set $\boldsymbol{\mathcal{PO}}_2^\alpha$ is composed only by $\mathcal{S}_m$ and $\mathcal{S}_c$.*

Proof. Let $w_0 \in W$ be the O/D pair which satisfies that $w_0 = \arg\max_{w \in W}\{\min\{d_{\mathcal{T}}(w), u^w\}\}$. We will assume that $d_{\mathcal{T}}(w_0) < u^w$ and $(\tilde{N}^{w_0}, \tilde{E}^{w_0}) \in \boldsymbol{\mathcal{T}}^\alpha$. Let us prove by *Reductio ad absurdum.* Thus, let us suppose that there exists $\mathcal{S} \in \boldsymbol{\mathcal{PO}}_2^\alpha$ such that $\mathcal{S} \neq \mathcal{S}_c$ and $\mathcal{S} \neq \mathcal{S}_m$. Therefore, $F_m(\mathcal{S}_m) < F_m(\mathcal{S}) < F_m(\mathcal{S}_c)$ and $F_c(\mathcal{S}_c) < F_c(\mathcal{S}) < F_c(\mathcal{S}_m)$, since if $F_m(\mathcal{S}) = F_m(\mathcal{S}_c)$ or $F_c(\mathcal{S}) = F_c(\mathcal{S}_m)$ then $\mathcal{S} \notin \boldsymbol{\mathcal{PO}}_2^\alpha$. With regard to $F_c(\mathcal{S}_c) < F_c(\mathcal{S}) < F_c(\mathcal{S}_m)$, it means that

$$\min\{d_{\mathcal{S}_c}(w_0), u^{w_0}\} < \min\{d_{\mathcal{S}}(w_0), u^{w_0}\} < \min\{d_{\mathcal{S}_m}(w_0), u^{w_0}\}.$$

In order to satisfy the previous inequality, we have to set

$$\min\{d_{\mathcal{S}_c}(w_0), u^{w_0}\} = d_{\mathcal{S}_c}(w_0) \text{ and } \min\{d_{\mathcal{S}}(w_0), u^{w_0}\} = d_{\mathcal{S}}(w_0).$$

Then,

$$d_{\mathcal{T}}(w_0) \leq d_{\mathcal{S}_c}(w_0) < d_{\mathcal{S}}(w_0) < \min\{d_{\mathcal{S}_m}(w_0), u^{w_0}\},$$

which is not possible since in a tree network the path for each O/D pair $w \in W$ is unique, since there are no cycles. Note that if $d_{\mathcal{T}}(w_0) \geq u^w$, then $F_c(\mathcal{S}_c) = F_c(\mathcal{S}) = F_c(\mathcal{S}_m) = u^{w_0}$. Besides, since $F_m(\mathcal{S}_m) < F_m(\mathcal{S})$ and $F_m(\mathcal{S}_m) < F_m(\mathcal{S}_c)$, then $\mathcal{S}, \mathcal{S}_c \notin \boldsymbol{\mathcal{PO}}_2^\alpha$. □

As in Ogryczak (1997), with the purpose of identifying some compromise λ-cent-dians on a general network, we need a solution concept different from the one discussed so far, which is an extension of the Halpern's λ-cent-dian concept in Location Theory (see Halpern (1978)). As explained for the case of a non-convex problem in Location Theory (see Steuer (1986)), the set of Pareto-optimality with respect to the bicriteria center/median objective can be completely parameterized with minimization of the weighted Chebychev norm. In the Network Design area, the Pareto-optimality set $\boldsymbol{\mathcal{PO}}_2^\alpha$ can be completely parameterized with minimization of the weighted function

$$\bar{H}_\lambda(\mathcal{S}) = \max\{\lambda\, F_c(\mathcal{S}), (1-\lambda)\, F_m(\mathcal{S})\}. \tag{5.17}$$

In the case of a non-unique optimal solution, this optimization has to be subject to some refinement. Thus, we call a subgraph $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ a *maximum λ-cent-dian* if it is an optimal solution to the following lexicographic (two-level) problem

$$\operatorname{lex}\min\{[\bar{H}_\lambda(\mathcal{S}), H_\lambda(\mathcal{S})] : \ \mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha\}. \tag{5.18}$$

The lexicographic minimization in (5.18) means that first we minimize $\bar{H}_\lambda(\,\cdot\,)$ on set $\boldsymbol{\mathcal{N}}^\alpha$, and then we minimize $H_\lambda(\,\cdot\,)$ on the optimal subnetworks set of function $\bar{H}_\lambda(\,\cdot\,)$. Thus, function $H_\lambda(\,\cdot\,)$, defined as the linear convex combination (5.12), is minimized only for regulation purposes, in the case of a nonunique minimum solution for the main function $\bar{H}_\lambda(\,\cdot\,)$. If the optimal solution is not unique, this regularization is necessary to guarantee that the maximum λ-cent-dian always belongs to $\boldsymbol{\mathcal{PO}}_2^\alpha$. Besides, each subgraph $\mathcal{S} \in \boldsymbol{\mathcal{PO}}_2^\alpha$ can be found as a maximum λ-cent-dian with $0 \leq \lambda \leq 1$. The proofs of these last two statements are detailed below in Propositions 8 and 9. By construction of the functions used, they are similar to the ones exposed for Propositions 3 and 4 in Ogryczak (1997) for Location Theory.

Proposition 8. *On any network $\mathcal{N}$, for each $\lambda \in (0,1)$, the corresponding maximum λ-cent-dian belongs to $\boldsymbol{\mathcal{PO}}_2^\alpha$.*

Proof. Let $\mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$ be a maximum λ-cent-dian for some $\lambda \in (0,1)$. Let us prove by *Reductio ad absurdum.* That is, suppose that $\mathcal{S} \notin \boldsymbol{\mathcal{PO}}_2^\alpha$. This means that there exists $\mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$ such that

$$F_c(\mathcal{S}') \leq F_c(\mathcal{S}) \quad \text{and} \quad F_m(\mathcal{S}') \leq F_m(\mathcal{S}),$$

where at least one of the inequalities is satisfied strictly. Hence, since $\lambda \in (0,1)$,

$$\bar{H}_\lambda(\mathcal{S}') \leq \bar{H}_\lambda(\mathcal{S}) \quad \text{and} \quad H_\lambda(\mathcal{S}') \leq H_\lambda(\mathcal{S}),$$

which contradicts the fact that $\mathcal{S}$ is a maximum λ-cent-dian. Thus, a maximum λ-cent-dian solution network always belongs to $\boldsymbol{\mathcal{PO}}_2^\alpha$. □

Proposition 9. *On any network $\mathcal{N}$, for each $\mathcal{S} \in \boldsymbol{\mathcal{PO}}_2^\alpha$ there exists $\lambda \in (0,1)$ such that $\mathcal{S}$ is the corresponding maximum λ-cent-dian.*

Proof. Let us consider $\mathcal{S} \in \boldsymbol{\mathcal{PO}}_2^\alpha$ and $\lambda = F_m(\mathcal{S})/(F_c(\mathcal{S}) + F_m(\mathcal{S}))$. Observe that $\lambda \in (0,1)$ and $1 - \lambda = F_c(\mathcal{S})/(F_c(\mathcal{S}) + F_m(\mathcal{S}))$. Then,

$$\bar{H}_\lambda(\mathcal{S}) = F_c(\mathcal{S})F_m(\mathcal{S})/(F_c(\mathcal{S}) + F_m(\mathcal{S})) = \lambda F_c(\mathcal{S}) = (1-\lambda)F_m(\mathcal{S}). \tag{5.19}$$

Let us prove it by *Reductio ad absurdum.* Let us suppose that $\mathcal{S}$ is not the corresponding maximum λ-cent-dian. This means that there exists $\mathcal{S}' \in \boldsymbol{\mathcal{N}}^\alpha$ such that

$$\bar{H}_\lambda(\mathcal{S}') \leq \bar{H}_\lambda(\mathcal{S}) \quad \text{and} \quad H_\lambda(\mathcal{S}') \leq H_\lambda(\mathcal{S}),$$

where at least one of the inequalities is satisfied strictly. That is,

$$\lambda F_c(\mathcal{S}') \leq \bar{H}_\lambda(\mathcal{S}) \quad \text{and} \quad (1-\lambda)F_m(\mathcal{S}') \leq \bar{H}_\lambda(\mathcal{S}), \tag{5.20}$$

where at least one of the inequalities has to be satisfied strictly. By equation (5.19), it would mean that $\mathcal{S} \notin \boldsymbol{\mathcal{PO}}_2^\alpha$. Thus, $\mathcal{S}$ has to be the corresponding maximum λ-cent-dian. □

Regarding Example 5, we have shown that the subnetwork $\mathcal{S}_1$ cannot be a λ-cent-dian for any $0 \leq \lambda \leq 1$ since $H_\lambda(\mathcal{S}_m) < H_\lambda(\mathcal{S}_1)$ or $H_\lambda(\mathcal{S}_c) < H_\lambda(\mathcal{S}_1)$. Note that $\bar{H}_\lambda(\mathcal{S}_1) = \max\{80\,\lambda, 56.52\,(1-\lambda)\}$, $\bar{H}_\lambda(\mathcal{S}_c) = \max\{70\,\lambda, 65.21\,(1-\lambda)\}$ and $\bar{H}_\lambda(\mathcal{S}_m) = \max\{92\,\lambda, 21.39\,(1-\lambda)\}$. Hence, $\bar{H}_\lambda(\mathcal{S}_1) < \bar{H}_\lambda(\mathcal{S}_c)$ and $\bar{H}_\lambda(\mathcal{S}_1) < \bar{H}_\lambda(\mathcal{S}_m)$ for any $0.4467 < \lambda < 0.4491$. In fact, subnetwork $\mathcal{S}_1$ is the maximum λ-cent-dian as long as $0.4467 < \lambda < 0.4491$.

As a conclusion, similar to the λ-cent-dian concept, the maximum λ-cent-dian generates the solution network depending on the value of $\lambda \in (0,1)$. As proven in Propositions 8 and 9, the difference between both concepts is that the maximum λ-cent-dian allows us to model all rational existing compromises between $F_c(\,\cdot\,)$ and

$F_m(\cdot)$. In the following example, we identify the $\boldsymbol{\mathcal{PO}}_2^\alpha$ set for a given network. Anyway, we have seen that getting the set $\boldsymbol{\mathcal{PO}}_2^\alpha$ has a high computational burden.

Example 5 cont Regarding the instance of Example 5 but considering $\alpha\, C_{total} = 90$ we have generated the whole set of $\mathcal{N}^\alpha$. Each point in Figure 5.6 represents a feasible solution from $\mathcal{N}^\alpha$. That is, solutions whose construction cost is less or equal to $\alpha\, C_{total}$. We have highlighted in red the only three solutions that are not dominated. They form the $\mathcal{PO}_2^\alpha$ set. Their construction cost is larger than or equal to the ones that are dominated. They are the corresponding maximum λ-cent-dians for $\lambda \in [0.0001, 0.3805]$, $\lambda \in [0.3806, 0.4354]$ and $\lambda \in [0.4355, 0.9999]$.

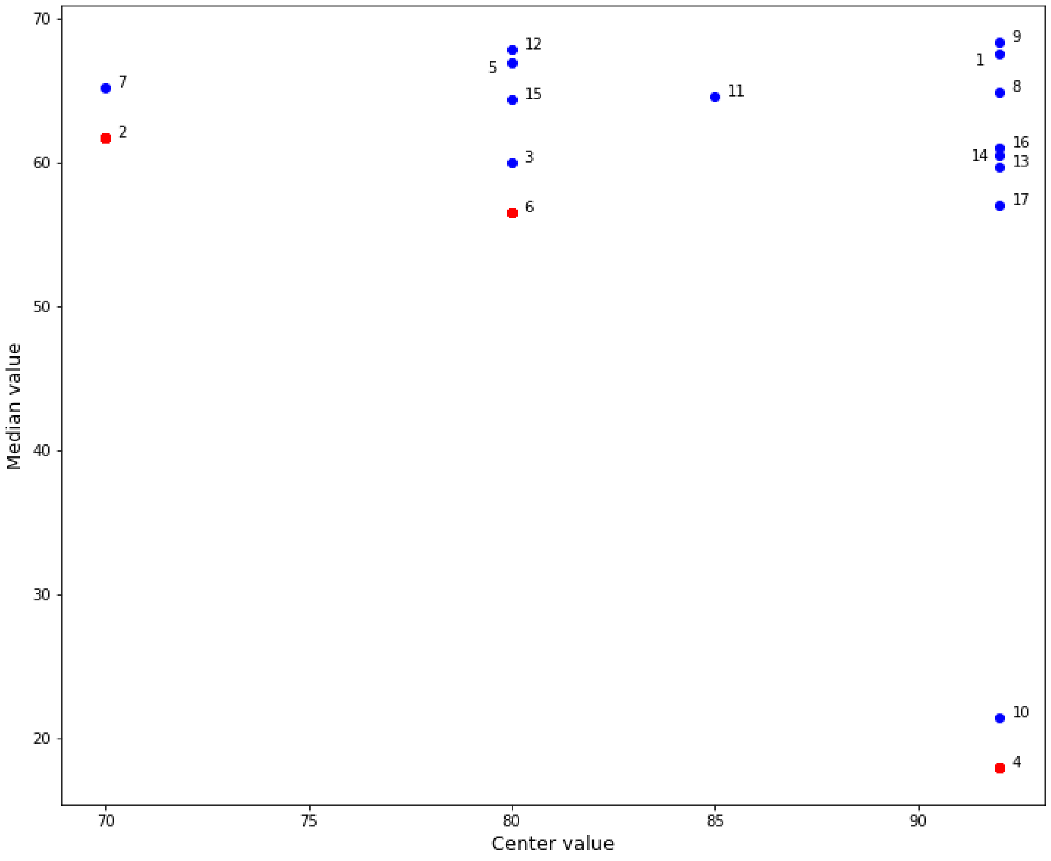


Figure 5.6: Identification of Pareto-optimal solutions for a given network.

Observing the whole set of feasible solutions we identify that solutions labeled 2 and 7 are both center solutions. The difference is that in solution 2 we are requiring a little percentage of efficiency with constraint (5.21). As this percentage grows, we will consequently obtain solution points 3, 6, 10 and 4. With respect to median solutions, point 4 is the most efficient one. Point 10 corresponds to a median solution when the budget is strictly less than 4. Regarding the generalized-center solution, point 7 corresponds to the solution that has the smallest difference between the values of the center and the median. In this case, the generalized-center corresponds with the less efficient center. We can require also a percentage of efficiency for the generalized-center solution. As this percentage grows, we will consequently obtain solutions points 2, 3, 6, 10 and 4. The rest of the points are solution networks for the three concepts studied when the budget is strictly less than the available budget.

5.2.3 Adding efficiency to the generalized-center problem. Evaluating some other inequality measures

We have already seen why the solutions of the generalized-center are cataloged as "weird" concerning the center and median functions. Furthermore, if we restrict the search of the generalized-center to the $\mathcal{PO}_2^\alpha$ set, it coincides with the least efficient center network. Anyway, if we want to "improve" the generalized-center solution with respect to the notion of efficiency, we can consider adding the constraint

$$F_m(\mathcal{S}) \leq (1+\Delta)\, F_m(\mathcal{S}_m), \qquad \text{with } \Delta \in [0,1]. \tag{5.21}$$

If we fix $\Delta = 0$ we are demanding that the median objective value in $\mathcal{S}$ has exactly the same value as in $\mathcal{S}_m$. Remember that by definition, $F_m(\mathcal{S}_m) \leq F_m(\mathcal{S}), \forall\, \mathcal{S} \in \boldsymbol{\mathcal{N}}^\alpha$. If $\Delta \in (0,1)$, we are giving up a percentage of efficiency. That is, the median objective value in the optimal solution network is allowed to be a certain percentage greater than that in the median network. Finally, if $\Delta = 1$, there is no requirement for the optimal solution related to efficiency.

Until now, to measure the quality of the solutions and to form the Pareto-optimality set, we have used the two most popular measures in the literature: the center and the median values. There exist inequality measures, such as the mean absolute difference, the mean absolute deviation and the Gini coefficient useful to evaluate the quality of a solution (see Mesa et al. (2003)). Taking them into account, the generalized-center solution subject to constraint (5.21) is not necessarily dominated by any λ-cent-dian solution with $\lambda \in [0,1]$.

We illustrate the impact of adding constraint (5.21) on the quality of the solution and the consideration of other inequality measures in Section 5.5.2.

5.3 Problem formulations

Problems in this chapter are NP-hard. Extensive Facility Location problems can be viewed as particular cases of Network Design where only the distance from the origin w^s of each O/D pair to the facility is considered (the distance through the subnetwork and the distance from the subnetwork to the destination w^t are zero).

In Puerto et al. (2018), it is explained that the search of the λ-cent-dian subtree on a tree network is NP-hard, considering that tips of the subtree are nodes of the underlying network, as in our case. Furthermore, in Schmidt and Schöbel (2014), the authors present a problem, in which, given a network, a set of O/D pairs and a factor $\alpha \in [0, 1)$, the goal is to design a length/budget-constrained subnetwork along which the travel costs are reduced by α, minimizing the worst case (center objective) cost of all routing requests. They proved that this problem, which is a similar case of the λ-cent-dian problem described in this chapter with $\lambda = 1$, is NP-hard. Hence, problems in this work are NP-hard.

In this section, we present mixed-integer linear formulations for the λ-cent-dian problem defined in the previous section as the minimization of equation (5.12). First, we present a mixed-integer formulation for the specific case $\lambda \in [0, 1]$. Then, we cast the general case as a bilevel problem formulation. This bilevel formulation is useful and necessary only for $\lambda > 1$. In particular, this bilevel formulation takes into account the case $\lambda \to \infty$, which corresponds with a valid formulation for the generalized-center problem (5.11). Using standard linearization techniques we transform the bilevel formulation into a single-level formulation. Furthermore, the condition to add efficiency to the generalized-center problem of equation (5.21) is considered in our proposed formulations. The generalized-center solution concept has not been discarded since, as explained above and shown in Section 5.5.2, the generalized-center solution is not dominated by the λ-cent-dian solutions, with $\lambda \in [0, 1]$ if different measures from the center and median functions are used to measure the quality. Finally, we also describe a procedure to get the set $\mathcal{PO}_2^\alpha$ based on solving two linear formulations.

According to the elements defined in Subsection 1.2.1, for both mixed-integer linear formulations presented in this chapter, almost the same binary variables than for Covering problems in Chapter 2 are used (see Section 2.2).

- Node selection variables. For each $i \in N$, y_i is a binary variable to decide whether or not node i is built.
- Edge selection variables. For each $e \in E$, x_e is a binary variable to decide whether or not edge e is built.
- Flow variables. For each $w \in W$, they are used to model a path from w^s to w^t in the network to build, if possible. Variable $f_a^w, a \in A$ takes value 1 if arc a belongs to such path for w.
- For each $w \in W$, we consider an extra artificial arc $r = (w^s, w^t)$ and its corresponding flow variable f_r^w to model the alternative mode. Variable f_r^w takes value 1 if the shortest path from w^s to w^t in the designed subgraph is

larger than u^w and 0 otherwise. In other words, $f_r^w = 1$ represents the binary decision of the demand taking the alternative mode. This extra flow variable represents a situation closely linked to that of variable z^w used in (MC) and (PC) formulations.

- Maximum distance path variable. The continuous variable γ^{max} takes the value of the maximum distance of any O/D pair in the solution network.

5.3.1 Mixed-Integer Linear formulation for $\lambda \in [0, 1]$

We show a formulation of the λ-cent-dian problem for the case $\lambda \in [0, 1]$, named as (CD).

$$
\begin{aligned}
(CD) \quad & \min \ \lambda\gamma^{max} + (1-\lambda)\frac{1}{G_{total}} \sum_{w\in W} g^w \left(\sum_{a\in A^w} d_a\, f_a^w + u^w\, f_r^w \right) && && (5.22)\\
& \text{s.t.} \sum_{e\in E} c_e\, x_e + \sum_{i\in N} b_i\, y_i \leq \alpha\, C_{total}, && && (5.23)\\
& x_e \leq y_i, && e \in E,\ i \in e, && (5.24)\\
& \sum_{a\in\delta_w^+(i)} f_a^w - \sum_{a\in\delta_w^-(i)} f_a^w = \begin{cases} 1, & \text{if } i = w^s,\\ 0, & \text{otherwise,}\end{cases} && w \in W,\ i \in N, && (5.25)\\
& f_a^w + f_{\hat{a}}^w \leq x_e, && w \in W,\ a = (i,j) \in A^w : e = \{i,j\}, && (5.26)\\
& \sum_{a\in A} d_a\, f_a^w + u^w\, f_r^w \leq \gamma^{max}, && w \in W, && (5.27)\\
& x_e,\ y_i, && i \in N,\ e \in E, && (5.28)\\
& f_a^w \in \{0,1\}, && w \in W,\ a \in A^w \cup \{r\}, && (5.29)\\
& \gamma^{max} \geq 0. && && (5.30)
\end{aligned}
$$

The objective function (5.22) to be minimized represents a convex combination between the center and the median objectives. Constraint (5.23) limits the total construction cost, being $\alpha \in (0, 1]$. If we consider $\alpha = 0$ the optimal solution network is to construct nothing, no O/D pair is served. Constraint (5.24) ensures that if an edge is constructed, then its terminal nodes are constructed as well. For each pair w, expressions (5.25), (5.26) and (5.27) guarantee demand conservation and link flow variables f_a^w with the extra flow variable f_r^w, the maximum distance path variable γ^{max} and design variables x_e. Constraints (5.26) are named *capacity constraints* and they force each edge to be used in at most one direction by each O/D pair. Constraints (5.27), referenced as *maximum distance constraints*, put an upper bound on the length of the path for each pair $w = (w^s, w^t)$. The form of the objective function ensures variable f_r^w to take value 0 only if there exists a path between w^s and w^t with length at most u^w in the prospective network. This path is represented by variables f_a^w. Then, γ^{max} represents the largest $\ell_{\mathcal{S}}(w)$ within set W. Finally, constraints (5.28), (5.29) and (5.30) state that variables are binary, except for variable γ^{max}.

Remark 12. *Constraints* (5.26) *can be replaced by*

$$f_a^w \leq x_e, \qquad w \in W,\, a = (i,j) \in A : e = \{i,j\},$$

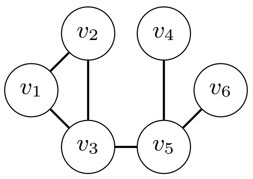
but the first one is stronger.

Remark 13. *Note that if $\alpha = 1$, there is not a limit of budget to invest. Then, the problem is a searching of shortest paths.*

For $\lambda \in [0,1)$, the objective function of the above formulation is composed of two positive coefficients (λ and $(1-\lambda)$) that multiply increasing functions of travel times. In consequence, at any optimal solution, the demand always chooses the shortest path without the necessity of enforcing this condition. That is, optimal solution vectors $\boldsymbol{f}^w$ correspond to shortest paths in the solution network. When $\lambda = 1$, this property does not hold, but the optimal solutions of the formulation are indeed centers. That is, the formulation above is still valid to compute center networks.

For the case $\lambda > 1$, the formulation (CD) is not valid anymore. Given that the term $(1-\lambda)$ is negative, each pair w will take the longest path between w^s to w^t in the designed network. We illustrate this situation in Example 6.

Example 6 Let us consider same potential network and its associated data from Example 3. Then, for $\alpha\, C_{total} = 100$ and $\lambda = 50 \cdot 10^4$, one of the optimal λ-cent-dian solution networks for formulation (CD) is



with the flow vector $f_r^w = 1$, $w \in \{(1,2),(4,1)\}$ and $f_r^w = 0$, $w \in \{(2,6)\}$. The path selected for $w = (2,6)$ is $[2,1,3,5,6]$ but the path $[2,3,5,6]$ is shorter than the selected one. Besides, this solution does not assign the pairs $(1,2)$ and $(4,1)$ to the existing paths in the network although they are shorter than the utilities $u^{(1,2)} = 92$ and $u^{(4,1)} = 92$.

5.3.2 Bilevel formulation for $\lambda > 1$. General problem formulation

We present the following bilevel formulation to impose that each $w \in W$ takes the shortest path in the solution network for any value of $\lambda \geq 0$. We call it the *bilevel*

λ-*cent-dian* formulation which is denoted as *(BCD)*.

$$(BCD) \quad \min \lambda\gamma^{max} + (1-\lambda)\frac{1}{G_{total}} \sum_{w\in W} g^w \left(\sum_{a\in A^w} d_a\, f_a^w + u^w\, f_r^w \right) \tag{5.31}$$

$$\text{s.t.} \sum_{e\in E} c_e\, x_e + \sum_{i\in N} b_i\, y_i \le \alpha\, C_{total}, \tag{5.32}$$

$$x_e \le y_i, \qquad e\in E,\, i\in e, \tag{5.33}$$

$$\sum_{a\in A} d_a\, f_a^w + u^w\, f_r^w \le \gamma^{max}, \qquad w\in W, \tag{5.34}$$

$$\boldsymbol{f}^w \in (F)^w, \qquad w\in W, \tag{5.35}$$

$$x_e,\, y_i,\, f_a^w \in \{0,1\}, \qquad i\in N,\, e\in E,\, a\in A\cup\{r\},\, w\in W, \tag{5.36}$$

$$\gamma^{max} \ge 0, \tag{5.37}$$

where

$$(F)^w = \arg\min_{\boldsymbol{h}} \left\{ \left(\sum_{a\in A^w} d_a\, h_a^w + u^w\, h_r^w \right) : \text{ s.t.} \begin{cases} \text{const.}(5.25)-(5.26), \\ h_a^w \in \{0,1\}, \\ a\in A\cup\{r\} \end{cases} \right\}. \tag{5.38}$$

The bilevel formulation (BCD) consists of an upper-level task that minimizes the λ-cent-dian function, which is composed of travel times of set W, and a lower-level task per each O/D pair $w\in W$, which is a search of the shortest path between w^s and w^t in the designed network. These lower-level problems are stated in equation (5.38), named $(F)^w$.

We reformulate the bilevel problem (BCD) as a single-level problem by imposing the optimality conditions of each problem $(F)^w$. To do so, we consider the dual of each problem $(F)^w$, denoted by $(DF)^w$.

$$(DF)^w \quad \max_{\boldsymbol{\phi},\boldsymbol{\sigma},\boldsymbol{v}} \quad \phi_{w^s} - \sum_{e\in E} x_e\, \sigma_e \tag{5.39}$$

$$\text{s.t.} \quad \phi_i - \phi_j - \sigma_e \le d_a, \qquad a=(i,j)\in A : e=\{i,j\}, \tag{5.40}$$

$$\phi_{w^s} \le u, \tag{5.41}$$

$$\sigma_e \ge 0, \qquad e\in E. \tag{5.42}$$

As it is clear from the context, we omit the index w. Variables $\phi_i, i\in N$, are the dual variables related to the flow constraints (5.25) and σ_e, $e\in E$, are the dual variables corresponding to the capacity constraints (5.26). Given that the set of flow constraints contains one linear dependent constraint, we set $\phi_{w^t}=0$.

By *strong duality*, a feasible vector $\boldsymbol{f}^w$ for $(F)^w$ is optimal if and only if there exist feasible vectors $\boldsymbol{\phi}$ and $\boldsymbol{\sigma}$ for $(DF)^w$ such that:

$$\sum_{a\in A} d_a\, f_a + u\, f_r = \phi_{w^s} - \sum_{e\in E} x_e\, \sigma_e.$$

Then, (BCD) can be cast as a single-level non-linear optimization problem

$$
\begin{aligned}
(BCD)\quad & \min \lambda\gamma^{max} + (1-\lambda)\frac{1}{G}\sum_{w\in W} g^w \left(\sum_{a\in A^w} d_a\, f_a^w + u^w\, f_r^w\right) && && (5.43)\\
& \text{s.t. } \sum_{e\in E} c_e\, x_e + \sum_{i\in N} b_i\, y_i \le \alpha\, C_{total}, && && (5.44)\\
& x_e \le y_i, && e\in E,\ i\in e, && (5.45)\\
& \sum_{a\in A} d_a\, f_a^w + u^w\, f_r^w \le \gamma^{max}, && w\in W, && (5.46)\\
& \sum_{a\in\delta_w^+(i)} f_a^w - \sum_{a\in\delta_w^-(i)} f_a^w = \begin{cases} 1, & \text{if } i = w^s,\\ 0, & \text{if } i\notin\{w^s,w^t\},\end{cases} && w\in W,\ i\in N, && (5.47)\\
& f_a^w + f_{\hat a}^w \le x_e, && w\in W,\ e=\{i,j\}\in E^w : a=(i,j), && (5.48)\\
& \phi_i^w - \phi_j^w - \sigma_e^w \le d_a, && w\in W,\ a=(i,j)\in A^w : e=\{i,j\}, && (5.49)\\
& \phi_{w^s}^w \le u^w, && w\in W, && (5.50)\\
& \sum_{a\in A^w} d_a\, f_a^w + u^w\, f_r^w = \phi_{w^s}^w - \sum_{e\in E^w} x_e\, \sigma_e^w, && w\in W, && (5.51)\\
& x_e,\ y_i,\ f_a^w \in\{0,1\}, && i\in N,\ e\in E,\ a\in A\cup\{r\},\ w\in W, && (5.52)\\
& \sigma_e^w \ge 0, && e\in E,\ w\in W, && (5.53)\\
& \gamma^{max}\ge 0. && && (5.54)
\end{aligned}
$$

To obtain a linear model we get rid of the products of variables present in (5.51). We introduce the set of variables ξ_e^w, representing the product $x_e\,\sigma_e^w$ and we linearize them by using the following McCormick inequalities (see McCormick (1976)):

$$\xi_e \le x_e\Sigma_e, \tag{5.55}$$

$$\xi_e \le \sigma_e, \tag{5.56}$$

$$\xi_e \ge \sigma_e - \Sigma_e(1-x_e), \tag{5.57}$$

$$\xi_e \ge 0. \tag{5.58}$$

where Σ_e is an upper bound of σ_e.

In the following, we consider the assumption that for each $w\in W$, $d_{\mathcal{N}}(w)\le u^w$. This is not a restrictive assumption since a pair not satisfying this condition will always prefer the private mode and can be eliminated from the analysis.

Proposition 10. *Given a potential network $\mathcal{N}(N,E)$, for every O/D pair $w\in W$ and every edge $e\in E$, $\Sigma_e = u^w - d_{\mathcal{N}}(w)$ is a valid bound for variable σ_e^w in (BCD).*

Proof. For each $w \in W$, rewriting (5.51), we have that

$$\sum_{e\in E^w} x_e\,\sigma_e^w = \phi_{w^s}^w - \left(\sum_{a\in A^w} d_a\,f_a^w + u^w\,f_r^w\right).$$

Using (5.50) and that $\sum\limits_{a\in A^w} d_a\,f_a^w + u^w\,f_r^w \geq d_{\mathcal{N}}(w)$, then

$$\sum_{e\in E^w} x_e\,\sigma_e^w \leq u^w - d_{\mathcal{N}}(w).$$

Thus, we can conclude that $\sigma_e^w \leq u^w - d_{\mathcal{N}}(w)$. □

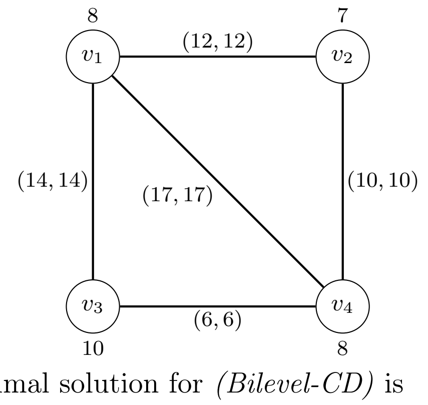
Proposition 11. *The bound given in Proposition 10 for each variable σ_e^w, $w \in W$, $e \in E$, of (BCD) formulation is the best one.*

Proof. Let us fix a pair $w \in W$ and consider the particular situation for which there exists $e = \{w^s, w^t\} \in E^w \subseteq E$ and $d_{\mathcal{N}}(w) = d_{(w^s,w^t)}$. Let us suppose that there exists a better bound for $\sigma_{\{w^s,w^t\}}^w$ than $< u^w - d_{\mathcal{N}}(w)$. That is, $\sigma_{\{w^s,w^t\}}^w \leq u^w - d_{\mathcal{N}}(w) - \epsilon$. By (5.49), $\phi_{w_s}^w \leq d_a + \sigma_{\{w^s,w^t\}}^w \leq u^w - \epsilon$. Then, according to (5.51),

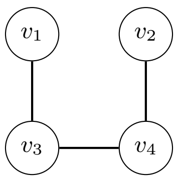
$$\phi_{w_s}^w = \sum_{a\in A^w} d_a\,f_a^w + u^w\,f_r^w + \sum_{e\in E^w} x_e\,\sigma_e^w \leq u^w - \epsilon. \tag{5.59}$$

In order to satisfy the previous inequality, f_r^w has to be set to 0, which forces a solution network to have a path for w (the corresponding flow variables take value equal to 1). This situation can result in an infeasible solution by (5.38) if the length of the assigned path is larger than its u^w or even there is no path.
Let us show it with a small example. Let $\mathcal{N}$ be the following potential network.

Origin	Destination	u^w	g^w
1	2	24	181
1	4	34	168
2	4	20	43
3	2	32	121



Being $\lambda = 20$ and $\alpha\,C_{total} = 63$, the optimal solution for *(Bilevel-CD)* is



which means that $x_{\{1,2\}} = 0$. By Proposition 11, it is set $f_r^{(1,2)} = 0$, which forces to assign w to the solution network. This solution network contains a path for w but it is larger than its u^w, which results in an infeasibility by (5.38) for not taking the optimal mode. □

Finally, as we have already explained, the generalized-center is a very inefficient solution and does not have to be a center network. In Section 5.2.3, we proposed a way to add efficiency to the generalized-center solution. That is, for $\lambda > 1$, we add to the proposed formulation (BCD) the constraint

$$\frac{1}{G_{total}} \sum_{w \in W} g^w \left(\sum_{a \in A^w} d_a\, f_a^w + u^w\, f_r^w \right) \leq (1 + \Delta) F_m(\mathcal{S}_m), \tag{5.60}$$

being $F_m(\mathcal{S}_m)$ the median objective value for the median network. That is, the objective value for (CD) with $\lambda = 0$.

5.3.3 Problem formulation to get the Pareto-optimal set

Subsection 5.2.2 proves that the set $\mathcal{PO}_2^\alpha$ can be completely parametrized as a function of $\lambda \in (0, 1)$ with minimization of the weighted function $\bar{H}_\lambda(\,\cdot\,)$. In the case of a non-unique optimal solution, we select the one that has minimum $H_\lambda(\,\cdot\,)$ value. That is, firstly, to minimize $\bar{H}_\lambda(\,\cdot\,)$ we solve the problem

$$\begin{aligned} \min \quad & \mu && (5.61)\\ \text{s.t.} \quad & \mu \geq \lambda\, F_c(\mathcal{S}), && (5.62)\\ & \mu \geq (1 - \lambda)\, F_m(\mathcal{S}), && (5.63)\\ & \mathcal{S} \in \mathcal{N}^\alpha, && (5.64)\\ & \mu \geq 0. && (5.65) \end{aligned}$$

Then, being v^* the objective value of the problem formulation (5.61)-(5.65), the following second problem is solved for regulation purposes:

$$\begin{aligned} \min \quad & \lambda\, F_c(\mathcal{S}) + (1 - \lambda)\, F_m(\mathcal{S}) && (5.66)\\ \text{s.t.} \quad & \lambda\, F_c(\mathcal{S}) \leq v^*, && (5.67)\\ & (1 - \lambda)\, F_m(\mathcal{S}) \leq v^*, && (5.68)\\ & \mathcal{S} \in \mathcal{N}^\alpha. && (5.69) \end{aligned}$$

In terms of sets N, E and W, the previous formulations are equivalent to the following ones.

$$
\begin{aligned}
(MCD\text{-}1)\ \min\quad & \mu && && (5.70)\\
\text{s.t.}\quad & \mu \geq \lambda \left(\sum_{a \in A^w} d_a\, f_a^w + u^w\, f_r^w \right), && w \in W, && (5.71)\\
& \mu \geq (1-\lambda)\, \frac{1}{G_{total}} \sum_{w \in W} g^w \left(\sum_{a \in A^w} d_a\, f_a^w + u^w\, f_r^w \right), && && (5.72)\\
& \mu \geq 0, && && (5.73)\\
& \text{constraints } (5.23), (5.24), (5.25), (5.26), (5.28). && && (5.74)
\end{aligned}
$$

$$
\begin{aligned}
(MCD\text{-}2)\ \min\ & \lambda \left(\sum_{a \in A^w} d_a\, f_a^w + u^w\, f_r^w \right) + (1-\lambda) \tfrac{1}{G_{total}} \sum_{w \in W} g^w \left(\sum_{a \in A^w} d_a\, f_a^w + u^w f_r^w \right) && && (5.75)\\
\text{s.t.}\ & 0 \leq v^* - \lambda \left(\sum_{a \in A^w} d_a\, f_a^w + u^w\, f_r^w \right), && w \in W, && (5.76)\\
& 0 \leq v^* - (1-\lambda)\, \frac{1}{G_{total}} \sum_{w \in W} g^w \left(\sum_{a \in A^w} d_a\, f_a^w + u^w\, f_r^w \right), && && (5.77)\\
& \mu \geq 0, && && (5.78)\\
& \text{constraints } (5.23), (5.24), (5.25), (5.26), (5.28). && && (5.79)
\end{aligned}
$$

5.4 Algorithmic discussion for $\lambda \in [0,1]$

Due to the NP-hardness of the λ-cent-dian problem, decomposition methods and metaheuristic approaches are developed and tested. Firstly, for $\lambda \in [0,1]$, we apply the same Benders decomposition approach than in Chapter 2 for covering problems, Subsection 2.3.3. It is based on the ideas of Conforti and Wolsey (2019) to generate facet-defining Benders cuts. Secondly, we present a GRASP algorithm, using the same ideas like the one designed for (MC) in García-Archilla et al. (2013). In Section 4.2, the GRASP procedure is detailed.

A preprocessing similar to that in Chapters 2 and 4 is suitable and fits well in this context, as justified below.

5.4.1 Preprocessing methods

A preprocessing similar to that for covering problems is appropriate in this context. Formulations *(CD)* and *(BCD)* do not have a set of mode choice constraints as *(MC)* and *(PC)*, but due to the nature of their objective functions an analogous situation happens: for each w, the minimum value of $\{d_{\mathcal{S}}(w), u^w\}$ will be selected, being $\mathcal{S}$ the subnetwork designed from the potential network $\mathcal{N}$. Then, the first part consists in building for each pair $w \in W$ a subgraph $\mathcal{N}^w$ composed solely of those nodes and edges that belong to any path from w^s to w^t shorter than or equal to u^w. Next, on the second part, we identify the set of O/D pairs which are too expensive to be served. That means, those pairs which do not have a path from w^s to w^t in $\mathcal{N}^w$ satisfying: i) its building cost is less than $\alpha\, C_{total}$; and ii) its length is less than u^w, are identified. This set is referenced as $\bar{W}$ and its decision about the mode choice is fixed, $f_r^w = 1$.

5.4.2 Benders decomposition for (CD) with $\lambda \in [0,1]$

Formulation (CD) involves a huge number of flow variables when the set of O/D pairs is large. Firstly, to circumvent this drawback with the procedure of Benders decomposition in an efficient way, we present some properties. Then, we describe the procedure to generate Benders cuts for (CD) based on the ideas exposed in Conforti and Wolsey (2019), as was done for (MC) and (PC) in Chapter 2. This implementation is used as a subroutine in the branch-and-Benders-cut scheme.

Proposition 12 shows that we can relax the integrality constraints on the flow variables f_a^w and f_r^w. Let (CD_R) denote the formulation (CD) in which constraints $f_a^w \in \{0,1\}, w \in W, a \in A \cup \{r\}$ are replaced by non-negativity constraints, i.e.

$$f_a^w \geq 0, w \in W, a \in A \cup \{r\}. \tag{5.80}$$

By Definition 1, the following proposition states.

Proposition 12. *The projections of (CD) and (CD_R) onto the $\boldsymbol{f}$-space coincide.*

$$Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD)) = Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD\_R)).$$

Proof. First, $\mathcal{F}(CD) \subseteq \mathcal{F}(CD\_R)$ implies $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD)) \subseteq Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD\_R))$. Second, let $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\gamma^{max}})$ be a point belonging to $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD\_R))$. Due to the structure of the objective function, every O/D pair will select one of the two modes, the shortest one. That is, f_r^w will take the value of 0 or 1. If $f_r^w = 1$ then $\boldsymbol{f}^w = 0$. In the case where $f_r^w = 0$, there exists a flow $f_a^w \geq 0$ satisfying (5.25) and (5.26) that can be decomposed into a convex combination of flows on paths from w^s to w^t and cycles, in the case that these paths have the same length. Given that the flow f_a^w also satisfies (5.27), then a flow of value 1 on one of the paths in the convex combination must satisfy this constraint. Hence, by taking f_a^w equal to 1 for the arcs belonging to this path and to 0 otherwise, we show that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\gamma^{max}})$ also belongs to $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD))$. $\square$

Based on Proposition 12, we propose a Benders decomposition where variables f_a^w and f_r^w are projected out from the model and replaced by Benders facet-defining cuts. Following the Benders decomposition Theory, since flow variables appear in the objective function of (CD), we have introduced an incumbent variable ζ^w for each O/D pair $w \in W$ to denote the expression $\sum_{a\in A^w} d_a\, f_a^w + u^w\, f_r^w$. Then, the master problem that we solve is

$$\begin{aligned}
(M\_CD) \quad \min \quad & \lambda\gamma^{max} + (1-\lambda)\frac{1}{G_{total}}\sum_{w\in W} g^w\zeta^w && (5.81)\\
\text{s.t.} \quad & \sum_{e\in E} c_e\, x_e + \sum_{i\in N} c_i\, y_i \le \alpha\, C_{total}, && (5.82)\\
& x_e \le y_i, \qquad e\in E,\ i\in\{e^s,e^t\}, && (5.83)\\
& \zeta^w \le \gamma^{max}, \qquad w\in W, && (5.84)\\
& x_e,\, y_i \in\{0,1\}, \quad \gamma^{max},\zeta^w \ge 0, \quad i\in N,\ e\in E,\ w\in W, && (5.85)
\end{aligned}$$

To generate facet-defining Benders cuts it is necessary to get an interior point of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}}}(\mathcal{F}(CD\_R))$. This convex hull is the same as for the projection which considers the incumbent variables ζ^w, $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}},\boldsymbol{\zeta}}(\mathcal{F}(CD\_R))$. Then, Proposition 13 gives us a way to get such interior point.

Proposition 13. *After preprocessing, the convex hull of* $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}},\boldsymbol{\zeta}}(\mathcal{F}(CD\_R))$ *is full-dimensional.*

Proof. To prove the result, we exhibit $|N|+|E|+|W|+2$ affinely independent feasible points:

- $y_i = 0,\ i\in N, \quad x_e = 0,\ e\in E, \quad \zeta^w = u^w,\ w\in W, \quad \gamma^{max} = \max_{w\in W}\{\zeta^w\}$.
- $y_i = 0,\ i\in N, \quad x_e = 0,\ e\in E, \quad \zeta^w = u^w,\ w\in W, \quad \gamma^{max} = 2\max_{w\in W}\{\zeta^w\}$.
- For each $i \in N$, the points:

$$y_i = 1,\ y_{i'} = 0,\ i'\in N\setminus\{i\}, \quad x_e = 0,\ e\in E, \quad \gamma^w = u^w,\ w\in W,$$
$$\gamma^{max} = \max_{w\in W}\{\zeta^w\}.$$

- For each $e \in E$, the point:

$$y_k = 1,\ k\in e,\ y_k = 0,\ k\in N\setminus\{i,j\}, \quad x_e = 1,\ x_{e'} = 0,\ e'\in E\setminus\{e\},$$
$$\zeta^w = u^w,\ w\in W, \quad \gamma^{max} = \max_{w\in W}\{\zeta^w\}.$$

- For each $w \in W$, the point:

$$y_i = 1,\ i\in \widetilde{N}^w,\ y_i = 0,\ i\in N\setminus\widetilde{N}^w, \quad x_e = 1,\ e\in\widetilde{E}^w,\ x_e = 0,\ e\in E\setminus\widetilde{E}^w,$$
$$\zeta^w = 2\,u^w,\ \zeta^{w'} = u^w,\ w'\in W\setminus\{w\}, \quad \gamma^{max} = \max_{w\in W}\{\zeta^w\}.$$

Then, we write them as rows of a matrix of dimension $(|N|+|E|+|W|+2) \times (|N|+|E|+|W|+1)$.

$$
\begin{array}{c|ccccccccccccccccccc|}
 & y_1 & y_2 & \dots & \dots & y_N & x_1 & x_2 & \dots & \dots & x_M & \zeta^1 & \zeta^2 & \dots & \dots & \zeta^W & \gamma^{max} \\
p_0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
p_1 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & 2\,a_1 \\
p_{i_1} & 1 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
p_{i_2} & 0 & 1 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
\vdots & \vdots & \vdots & \ddots & & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
\vdots & \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
p_{i_N} & 0 & 0 & \dots & \dots & 1 & 0 & 0 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
p_{e_1} & & & & & & 1 & 0 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
p_{e_2} & & & & & & 0 & 1 & \dots & \dots & 0 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
\vdots & & & & & & \vdots & \vdots & \ddots & & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
\vdots & & & & & & \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
p_{e_M} & & & & & & 0 & 0 & \dots & \dots & 1 & u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_1 \\
p_{w_1} & & & & & & & & & & & 2\,u^{w_1} & u^{w_2} & \dots & \dots & u^{w_W} & a_2 \\
p_{w_2} & & & & & & & & & & & u^{w_1} & 2\,u^{w_2} & \dots & \dots & u^{w_W} & a_3 \\
\vdots & & & & & & & & & & & \vdots & \vdots & \ddots & & \vdots & \vdots \\
\vdots & & & & & & & & & & & \vdots & \vdots & & \ddots & \vdots & \vdots \\
p_{w_W} & & & & & & & & & & & u^{w_1} & u^{w_2} & \dots & \dots & 2\,u^{w_W} & a_W
\end{array}
$$

To prove that they are affinely independent one of them is selected, p_0, and we prove that vectors $p_1 - p_0$, $p_{i_1} - p_0$, $p_{i_2} - p_0$, $\dots$, $p_{i_N} - p_0$, $p_{e_1} - p_0$, $p_{e_2} - p_0$, $\dots$, $p_{e_M} - p_0$, $p_{w_1} - p_0$, $p_{w_2} - p_0$, $\dots$, $p_{w_W} - p_0$ are linearly independent.

$$
\begin{array}{c|ccccccccccccccccccc|}
 & y_1 & y_2 & \dots & \dots & y_N & x_1 & x_2 & \dots & \dots & x_M & \zeta_1 & \zeta_2 & \dots & \dots & \zeta_W & \gamma^{max} \\
p_1 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & a_1 \\
p_{i_1} & 1 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 \\
p_{i_2} & 0 & 1 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
\vdots & \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
p_{i_N} & 0 & 0 & \dots & \dots & 1 & 0 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 \\
p_{e_1} & & & & & & 1 & 0 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 \\
p_{e_2} & & & & & & 0 & 1 & \dots & \dots & 0 & 0 & 0 & \dots & \dots & 0 & 0 \\
\vdots & & & & & & \vdots & \vdots & \ddots & & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
\vdots & & & & & & \vdots & \vdots & & \ddots & \vdots & \vdots & \vdots & \dots & \dots & \vdots & \vdots \\
p_{e_M} & & & & & & 0 & 0 & \dots & \dots & 1 & 0 & 0 & \dots & \dots & 0 & 0 \\
p_{w_1} & & & & & & & & & & & u^{w_1} & 0 & \dots & \dots & 0 & a_2 - a_1 \\
p_{w_2} & & & & & & & & & & & 0 & u^{w_2} & \dots & \dots & 0 & a_3 - a_1 \\
\vdots & & & & & & & & & & & \vdots & \vdots & \ddots & & \vdots & \vdots \\
\vdots & & & & & & & & & & & \vdots & \vdots & & \ddots & \vdots & \vdots \\
p_{w_W} & & & & & & & & & & & 0 & 0 & \dots & \dots & u^{w_W} & a_W - a_1
\end{array}
$$

It is easy to see that by elementary row operations using p_1 as pivot row a lower echelon form is obtained. Then, vectors are linearly independent. □

The proof of Proposition 13 gives us a way to compute an interior point of the convex hull of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}},\boldsymbol{\zeta}}(\mathcal{F}(CD\_R))$. The average of these $|N|+|E|+|W|+2$ points, is indeed such an interior point.

Given an interior point $(\boldsymbol{x}^{in}, \boldsymbol{y}^{in}, \boldsymbol{\gamma^{max}}^{in}, \boldsymbol{\zeta}^{in})$ of the convex hull of feasible solutions and an exterior point $(\boldsymbol{x}^{out}, \boldsymbol{y}^{out}, \boldsymbol{\gamma^{max}}^{out}, \boldsymbol{\zeta}^{out})$, that is a solution to the LP relaxation of the current restricted master problem, a cut that induces a facet or an improper face of the polyhedron defined by the LP relaxation of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}},\boldsymbol{\zeta}}\mathcal{F}(CD)$ is generated. We denote the difference $\boldsymbol{x}^{out} - \boldsymbol{x}^{in}$ by $\Delta\boldsymbol{x}$. We define $\Delta\boldsymbol{y}$, $\Delta\boldsymbol{\gamma^{max}}$ and $\Delta\boldsymbol{\zeta}$ analogously. The idea is to find the furthest point from the core point, feasible to the LP-relaxation of $Proj_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{\gamma^{max}},\boldsymbol{\zeta}}\mathcal{F}(CD)$ and lying on the segment line between the interior point and the exterior point. This point is of the form $(\boldsymbol{x}^{sep}, \boldsymbol{y}^{sep}, \boldsymbol{\gamma^{max}}^{sep}, \boldsymbol{\zeta}^{sep}) = (\boldsymbol{x}^{out}, \boldsymbol{y}^{out}, \boldsymbol{\gamma^{max}}^{out}, \boldsymbol{\zeta}^{out}) - \lambda(\Delta\boldsymbol{x}, \Delta\boldsymbol{y}, \Delta\boldsymbol{\gamma^{max}}, \Delta\boldsymbol{\zeta})$. The problem of generating such a cut reads as follows:

$$(SP)^w \ \min_{\boldsymbol{f},\mu} \mu \tag{5.86}$$

$$\text{s.t.} \sum_{a\in\delta^+(i)} f_a - \sum_{a\in\delta^-(i)} f_a = \begin{cases} 1, & \text{if } i = w^s, \\ 0, & \text{otherwise,} \end{cases} \qquad i \in N, \tag{5.87}$$

$$f_a + f_{a'} \le x_e^{out} - \mu\,\Delta x_e, \qquad e = \{i,j\} \in E : a = (i,j), a' = (j,i), \tag{5.88}$$

$$\sum_{a\in A^w} d_a\, f_a + u\, f_r \le \zeta^{out} - \mu\,\Delta\zeta, \tag{5.89}$$

$$0 \le \mu \le 1, \tag{5.90}$$

$$f_a \ge 0, \qquad a \in A \cup \{r\}. \tag{5.91}$$

In order to obtain the Benders feasibility cut we solve its associated dual:

$$(DSP)^w \quad \max_{\boldsymbol{\phi},\boldsymbol{\sigma},\boldsymbol{\upsilon}} \quad \phi_{w^s} - \sum_{e\in E} x_e^{out}\,\sigma_e - \zeta^{out}\,\upsilon \tag{5.92}$$

$$\text{s.t.} \quad -\sum_{e\in E} \Delta x_e\,\sigma_e - \Delta\zeta\,\upsilon \le 1, \tag{5.93}$$

$$\phi_i - \phi_j - \sigma_e - d_a\,\upsilon \le 0, \qquad a = (i,j) \in A : e = \{i,j\}, \tag{5.94}$$

$$\phi_{w^s} - u\,\upsilon \le 0, \tag{5.95}$$

$$\sigma_e \ge 0, \quad \upsilon \ge 0, \qquad e \in E. \tag{5.96}$$

Given that $(SP)^w$ is always feasible ($\mu = 1$ is feasible) and that its optimal value is lower bounded by 0, then, both $(SP)^w$ and $(DSP)^w$ have always finite optimal solutions. Whenever the optimal value of μ is 0, $(\boldsymbol{x}^{out}, \boldsymbol{y}^{out}, \boldsymbol{\gamma^{max}}^{out}, \boldsymbol{\zeta}^{out})$ is feasible.

A cut is added if the optimal value of $(DSP)^w$ is strictly greater than 0. The new cut has the form

$$-\sum_{e\in E}\sigma_e\, x_e - \upsilon\,\zeta \le -\phi_{w^s}. \tag{5.97}$$

Finally, a set of valid inequalities is proposed in Remark 14

Remark 14. *By projecting out variables **f**, information regarding the lower bound of variables ζ^w is lost. To deal with that, we use the following set of valid inequalities*

$$\zeta^w \ge \sum_{e\in \tilde{E}^w} d_e\, x_e \quad \textit{if } d_{\mathcal{N}}(w) \le u^w, \quad w\in W. \tag{5.98}$$

5.4.3 GRASP for (CD) with $\lambda \in [0,1)$

We consider the metaheuristic proposed by García-Archilla et al. (2013) in the context of covering network design problems, where the authors considered a GRASP (Greedy Randomized Adaptive Search Procedure) to find good quality solutions. Here we adapt these ideas to the λ-cent-dian objective.

Algorithm 10: Construction Phase

```
Initialization: $\bar{\mathcal{N}} = (\bar{N}, \bar{E})$ with $\bar{N} = \emptyset$ and $\bar{E} = \emptyset$, $\bar{W} = \emptyset$, $C = 0$.
Compute $FE = \{e = \{i,j\} \in E \setminus \bar{E} \mid c_e + b_i + b_j{}^1 \le \alpha\, C_{total} - C\}$
while $FE \neq \emptyset$ do
    if $|FE| > k$ then
        Determine a set $SE = \{e_1, \dots, e_k\}$ consisting of the $k$ edges on $FE$
          that generate the minimum λ-cent-dian-distance value when they
          are individually added to $\bar{E}$.
    else
        Set $SE = FE$
    end
    Randomly choose one edge $e^* = \{i^*, j^*\} \in SE$ to be added to $\bar{\mathcal{N}}$.
    Update $\bar{\mathcal{N}}$, $FE$, $C$.
end
for $w \in W$ do
    Compute the shortest path for $w$ in $\bar{\mathcal{N}} = (\bar{N}, \bar{E})$.
    if its length is equal to or shorter than $u^w$ then
        $\bar{W} = \bar{W} \cup \{w\}$
    end
end
return $\bar{\mathcal{N}}$, $\bar{W}$
```

[1] Cost $c_{i'}$ (respectively $c_{j'}$) is taken into account if $i' \notin \bar{N}$.

The GRASP procedure works as follows. First, a construction phase is executed, in which an initial solution is built from scratch. Then an improvement phase is computed to improve this initial solution. This randomized Local Search scheme is applied n_{iter_max} times and the best solution network is picked.

The construction phase consists of adding one edge per iteration to an empty initial graph respecting the budget constraint until no more edges can be added. At each iteration, the set of feasible edges is updated. An edge is said to be a *feasible edge* if it has not been built previously and its cost is equal to or lower than the remaining available budget. Next, from this set, the $k \geq 2$ edges that contribute the greatest minimization to the objective value when they are individually added to the network under construction are selected. Then, one of them is randomly chosen to be added to the network under construction. This procedure is repeated until the set of feasible edges is empty at some iteration reached. This routine is described in Algorithm 10.

Algorithm 11: Improvement phase

```
Initialization: $\bar{\mathcal{N}} = (\bar{N}, \bar{E})$, $\bar{W} = \emptyset$, $C_{imp} = C$.
Compute $\bar{E}^C = E \setminus \bar{E}$.
for $\bar{e} \in \bar{E}$ do
    Define $\bar{\mathcal{N}}_{\bar{e}} = (\bar{N}, \bar{E} \setminus \{\bar{e}\})$
    $B_{\bar{e}} = \alpha\, C_{total} - C_{imp} - c_{\bar{e}} - b_{\bar{i}} - b_{\bar{j}}$[2]
    Compute $FE_{\bar{e}} = \{e = \{i,j\} \in \bar{E}^C \setminus \bar{E} \,|\, c_e + b_i + b_j{}^2 \leq B_{\bar{e}}\}$
    while $FE_{\bar{e}} \neq \emptyset$ do
        Determine the edge $\tilde{e} = \{\tilde{i}, \tilde{j}\} \in FE_{\bar{e}}$ that gives the largest
          improvement in the objective function when added to $\bar{\mathcal{N}}_{\bar{e}}$.
        Update $\bar{\mathcal{N}}_{\bar{e}}$, $FE_{\bar{e}}, B_{\bar{e}}$.
    end
end
Set $\bar{\mathcal{N}}^*$ the network that gives the largest improvement in $H_\lambda(\,\cdot\,)$ among all
  $\bar{\mathcal{N}}_{\bar{e}}$, $\bar{e} \in \bar{E}$.
for $w \in W$ do
    Compute the shortest path for $w$ in $\bar{\mathcal{N}}^*$.
    if its length is equal to or shorter than $u^w$ then
        $\bar{W} = \bar{W} \cup \{w\}$
    end
end
return $\bar{\mathcal{N}}^*$, $\bar{W}$
```

The improvement phase works as follows. For each edge e in the current solution network, a new network is built by replacing e with a set of feasible non-built edges.

[2]Cost b_i (respectively b_j) is taken into account if $i \notin \bar{N}$ (respectively $j \notin \bar{N}$).

These edges are added using the same criterion of the construction phase. The iteration finishes by picking the best solution obtained and comparing it with the solution obtained in the construction phase. This procedure is depicted in Algorithm 11.

In Subsection 5.5.4, a computational study is dedicated to this GRASP.

5.5 Computational experiments

In this section, we evaluate the quality of the solutions for the different solution concepts described in Section 5.2 using some inequality measures. Besides, we tested the performance of the two techniques proposed in Section 5.4 to solve the problem of (CD) for the case $\lambda \in [0, 1]$.

We perform these computational experiments on a computer equipped with an Intel Core i5-7300 CPU processor, with 2.50 gigahertz 4-core, and 16 gigabytes of RAM. The operating system is 64-bit Windows 10. We implement our codes in Python 3.8. We carried out these experiments through `CPLEX 12.10` solver, named `CPLEX`, using its Python interface. `CPLEX` parameters were set to their default values and the models were optimized in a single-threaded mode. All the experiments have been performed with a limit of one hour of CPU time. Besides, the different statistical analyzes have been done through the use of `R`. For such experiments and analyzes, `t` denotes the average value for solution times given in seconds, `gap` denotes the average of relative optimality gaps in percent (the relative percent difference between the best solution and the best bound obtained within the time limit), `Value` denotes the best value for GRASP and `cuts` is the average of number of cuts generated by the branch-and-Benders-cut implementations.

5.5.1 Data sets: benchmark networks and random instances

The tested instances for the algorithms exposed in this chapter are also divided into two groups, such as in previous chapters: *benchmark networks* and *random instances*.

On the one hand, there are three benchmark networks considered and for all of them α is set to 0.5. The Seville city network instance (referenced as `Seville`) is used in the same manner as previously, with the exception that for each $w \in W$ its utility u^w is set to twice the Euclidean distance. For the Sioux Falls city network instance (referenced as `Sioux Falls`), the topology of the network is described by 24 nodes and 38 edges. Set W is also formed by all possible O/D pairs $(38 \cdot 37 = 1406)$. The parameters have been chosen in the same manner as for the random instances. Finally, we also tested our algorithms on the benchmark instance `Ta2` from SNDLib (`http://sndlib.zib.de/`). From `Ta2` instance we have only used the topology of the underlying network, the cost vector for the set of arcs and the demand matrix. Its

O/D pairs set has 61% of them with zero demand. The rest of the parameters have been chosen in the same manner as for the random instances used in this chapter.

On the other hand, the random instances are generated in the same manner as in the previous chapters with some different points. For this experiment, we use random instances of size $n = 60$ nodes, named as N60 in what follows. There are 10 underlying networks of this size. The underlying planar graph of each instance has been generated with higher density. That is, their edges are deleted with probability 0.2. The budget $\alpha\, C_{total}$ is equal to 50% of the cost of building the whole underlying graph considered, which means that $\alpha = 0.5$. Besides, to build the set of O/D pairs W, all possible ones are taken into account. Thus, set W is composed of $n(n-1)$ elements. For each $w \in W$, parameter u^w is set to 2 times the Euclidean length between w^s and w^t. The rest of the parameters of the instance are chosen as in the previous chapter.

5.5.2 Quality of the solutions with a numerical illustration

In this subsection, we evaluate the quality of the solutions for the different solution concepts described in Section 5.2 using some inequality measures.

The instance used in this part has been generated in the same random manner than those described in the previous subsection. For this numerical illustration, we have consider a network of size $n = 20$ nodes. We have set $\alpha\, C_{total}$ equal to 40% of the cost of building the whole underlying network considered. Then, $\alpha = 0.4$. The rest of the parameters are chosen as previously.

The quality of the solutions of λ-cent-dian and generalized-center problems is studied as a function of λ and Δ, using some inequality measures. We first solve the λ-cent-dian for $\lambda \in \{0, 0.25, 0.5, 0.75\}$. Then we consider the center problem ($\lambda = 1$) and the approximation of the generalized-center with $\lambda = 500$. For them, the effect of adding the efficiency constraint (5.21) with $\Delta \in \{0\%, 3\%, 5\%, 10\%, 15\%\}$ is taken into account. Note that if we fix $\lambda = 1$ and $\Delta = 0$, we are solving the inverse lexicographic problem to that of (5.16). To all these cases, we add the solution found by the lexicographic problem (5.16). To get these solution networks, we have solved formulations (CD) and (BCD) exposed in Section 5.3 with the direct use of `CPLEX`.

We analyze the quality of the different solutions by comparing their minimum travel time (Min in what follows), maximum travel time (Max), and average travel time (Average), as well as the mean absolute difference (M.A.Difference) of their travel times. We relate the average travel time with the efficiency notion (median value) and the maximum travel time with the fairness one (center value). The percentage of O/D pairs served is also computed. These results are shown in Table 5.6. To these results, we add the solution of the lexicographic problem (5.16) whose minimum and maximum travel times are 4 and 96, respectively. Its average travel time, mean absolute difference between the travel times and the percentage of the demand covered are 47.149, 12.376 and 27.149, respectively.

By observing Table 5.6, we obtain some conclusions about the effect of the different values given to λ and Δ. First, note that there are only two different solutions obtained by solving the problem for $\lambda \in (0, 1)$. Note that these solutions belong to the set $\mathcal{PO}_2^{\alpha}$. We observe that for $\lambda \in [0, 0.75]$, the larger it is, the larger the average of travel times and the smaller the maximum travel time and the percentage of O/D pairs served. For $\lambda \in \{1, 500\}$ and $\Delta = 0$, the formulation consists of searching the median solution with the minimum center and generalized-center value, respectively. If $\Delta > 0$, it is observed that the average of travel times is increasing, bigger than in the median solution (obviously) and smaller than in the generalized-center solution. Besides, using the efficiency constraint causes more O/D pairs to be served than in the case where it is not used, as long as parameter Δ is not too large. Moreover, for a fixed value of Δ, the percentage of the O/D pairs served by the generalized-center solution is bigger than or equal to the center solution. With respect to the mean absolute difference, we conclude that there is no clear trend. We observe that in this case, the center solution is the one with the lowest value of this measure.

λ	Δ (%)	Min	Max	Average	M.A.Difference	O/D pairs served (%)
0	-	4	100	45.037	12.351	28.187
0.25	-	4	100	45.037	12.351	28.187
0.5	-	4	96	47.149	12.375	27.516
0.75	-	4	96	47.149	12.375	27.516
1	-	4	96	47.383	12.187	6.375
1	0	4	100	45.037	12.351	28.187
1	3	4	100	45.981	12.322	24.832
1	5	4	96	47.149	12.375	23.154
1	10	4	96	47.383	12.187	8.389
1	15	4	96	47.383	12.187	8.389
500	-	4	96	48.655	12.535	21.476
500	0	4	100	45.037	12.351	28.187
500	3	5	100	46.386	12.279	24.832
500	5	4	96	47.283	12.859	26.174
500	10	4	96	48.655	12.535	21.476
500	15	4	96	48.655	12.535	21.476

Table 5.6: Quality of solutions for λ-cent-dian and generalized-center problems using a random instance from N20 with $\alpha = 0.4$.

Besides, we observe that comparing the center solutions between them, there are some dominated ones. More specifically, the center solution without the efficiency constraint is the one that is completely dominated by the one in which this constraint is used with $\Delta = 15$. On the other hand, comparing the generalized-centers between them, a similar situation occurs.

Making a general comparison, we highlight that there are some generalized-center solutions non-dominated by the rest. The generalized-center solution with $\Delta = 3$ has smaller values of M.A.D than any of λ-cent-dian solutions with $\lambda < 1$. If we compare it with the center solutions, we see that for some of the cases, the center solution has a smaller M.A.D value but a bigger Average value and a lower percentage of demand covered. If this does not happen, the center has a worse M.A.D value but has better values for the rest of inequality measures.

5.5.3 Branch-and-Benders-cut performance

Our preliminary experiments show that including cuts only at integer nodes of the *branch-and-bound* tree is more efficient than including them in nodes with fractional solutions. Thus, in our experiments, we only separate integer solutions unless we specify the opposite. We used the `LazyConstraintCallback` function of `CPLEX` to separate integer solutions. Fractional solutions were separated using the `UserCutCallback` function. We study the implementation of `B&BC` (branch-and-Benders-cut algorithm) proposed in Subsection 5.4.2. As in Chapter 2, we will use the nomenclature of `BD_CW` to refer to the `B&BC` algorithm which generates cuts in the form of equation (5.97).

We compare our `BD_CW` implementation with the direct use of `CPLEX` and with the automatic Benders procedure proposed by `CPLEX`, noted by `Auto_BD`. The latter has been used in the option in which `CPLEX` attempts to decompose the model strictly according to the decomposition provided by the user. Besides, equations (5.98) are added in `Auto_BD` and `BD_CW`.

As for covering problems studied in Chapter 2, we observe that our `BD_CW` implementation for (CD) formulation appears to be competitive against `Auto_BD` whenever the size of the network is considerably large. Tables 5.7 and 5.8 show results for some benchmark instances, varying the values of the parameters α and λ. We highlight the best configuration in each case. Considering as a medium size instance `Seville` network, with 49 nodes, we see that in some cases `Auto_BD` is faster than our `BD_CW` implementation. However, for instances with more than 60 nodes, `BD_CW` is competitive against `Auto_BD`, such as shown for the instance `Ta2`.

For instances smaller than the `Seville` network, neither `Auto_BD` nor `BD_CW` are competitive with the direct use of `CPLEX`. For that, see Table C.1 in Appendix C, for which the `Sioux Falls` instance has been used. Furthermore, we observe that the tighter the budget, the lower the computation time `t` and the `gap`. With respect to λ parameter, it seems that it is easier to solve the problem for $\lambda = 1$.

		CPLEX		Auto_BD			BD_CW		
α	λ	t	gap	t	gap	cuts	t	gap	cuts
0.15	0	3600	1.277	**1107.906**	0.007	3805	3600	1.585	34774
	0.25	3600	3.492	3600	**0.187**	12009	3600	2.189	23604
	0.5	3600	5.513	3600	2.749	13305	3600	**0.849**	18392
	0.75	3600	7.24	3600	2.421	8441	3600	**0.385**	18429
	1	1263.11	0	**156.234**	0	660	186.484	0	22360
0.25	0	3600	2.039	**1333.359**	0.01	4460	3600	1.344	38008
	0.25	3600	10.125	**3518.875**	0.010	16048	3600	1.043	39937
	0.5	3600	16.758	3600	2.088	19590	3600	**1.201**	47348
	0.75	3600	20.496	3600	2.379	10705	3600	**0.107**	51078
	1	578.75	0	**263.297**	0	521	398.047	0	40293
0.4	0	3600	2.988	**1920.234**	0.010	11273	3600	2.539	33993
	0.25	3600	16.990	3600	10.494	21984	3600	**5.982**	25338
	0.5	3600	25.952	3600	**3.235**	14064	3600	8.681	31368
	0.75	3600	30.282	3600	21.454	16537	3600	**8.899**	35231
	1	3600	6.136	**521.625**	0	884	1812.359	0.007	89470

Table 5.7: Sensitivity analysis of BD_CW implementation for formulation (CD) with Seville instance.

		CPLEX		Auto_BD			BD_CW		
α	λ	t	gap	t	gap	cuts	t	gap	cuts
0.15	0	659.109	0	242.422	0	1809	**210.109**	0	18012
	0.25	1430.906	0	357.328	0	3687	**323.844**	0	26442
	0.5	1527.859	0	526.344	0	3932	**187.172**	0	15877
	0.75	2184.25	0	508.234	0	4146	**338.297**	0	31347
	1	433.75	0	315.984	0	314	**131.688**	0	25011
0.25	0	536.75	0	669.484	0	1980	**227.016**	0	25224
	0.25	3217.172	0	367.313	0	3508	**287.516**	0	24061
	0.5	3329.797	0	541.063	0	3867	**390.891**	0	26097
	0.75	2745.969	0	1024.344	0	5085	**328.719**	0	31378
	1	255.797	0	453.859	0	496	**145**	0	26937
0.4	0	735.031	0	318.078	0	2220	**247.813**	0	32694
	0.25	1424.563	0	445.359	0	3707	**340.156**	0	24374
	0.5	1804.578	0	946.453	0	4319	**553.156**	0	32843
	0.75	3600	0.807	3600	0.182	8796	**658.484**	0	35272
	1	488.547	0	555.563	0	883	**197.297**	0	33306

Table 5.8: Sensitivity analysis of BD_CW implementation for formulation (CD) with the Ta2 instance.

5.5.4 GRASP performance

To measure the quality of the solutions obtained by GRASP, we carried out a sensitivity analysis for the set of random instances N60 described in Subsection 5.5.1. The explored variables for this computational study are:

- `Value`: The overall normalized objective value of the solution. We divide the objective value obtained with the GRASP algorithm by the objective value in the situation where nothing is built.
- `t`: The overall execution time.
- `RPD`: The average relative percent deviation with respect to the best solution found by the `BD_CW` implementation after 1 hour. It is computed by the following formula:
$$\mathtt{RPD} = 100\frac{Z_{\mathtt{BD\_CW}} - Z'}{Z_{\mathtt{BD\_CW}}}, \tag{5.99}$$
being Z' and $Z_{\mathtt{BD\_CW}}$ the values of the solutions obtained by the GRASP algorithm and the `BD_CW` implementation, respectively.

For this analysis, we consider all the possible combinations of the following values for each parameter on N60 instances.

- The stopping criteria: $n_{iter\_max} \in \{5, 10, 20\}$
- Cardinality of the set of feasible edges to select from:
$k \in \{1, n/20, n/10, n\}$

We fix n_{iter_max} equal to 1 whenever the value for k is 1 since there is not a random component in the metaheuristic. We do not consider values n_{iter_max} larger than 20 because GRASP takes more than 1 hour to get a solution.

We present the optimality gap for the `BD_CW` implementation after 1 hour in Table 5.9. The greater the weight given to the center objective, the more difficult it is to solve the problem. In Table 5.10, we report averages for solution time and normalized objective value of GRASP for the different values of the parameters n_{iter_max} and k. Besides, the `RPD` is shown. Moreover, Table 5.11 collects the best situation for which there is no random component in GRASP ($k = 1$).

λ			
0	0.25	0.5	0.75
4.87	24.96	26.52	29.70

Table 5.9: `BD_CW` implementation gap for N60 after 1 hour.

λ	k	n_{iter_max}								
		5			10			20		
		`t`	`Value`	`RPD`	`t`	`Value`	`RPD`	`t`	`Value`	`RPD`
0	2	1108.196	0.842	-3.516	2191.655	0.843	-3.708	3600	0.841	-3.401
	15	916.186	0.891	-9.605	1947.392	0.882	-8.589	3600	0.872	-7.189
	60	715.623	0.934	-14.899	1317.576	0.916	-12.706	2604.273	0.911	-12.105
0.25	2	1097.559	0.867	12.132	2180.521	0.878	11.015	3600	0.877	11.147
	15	952.412	0.936	5.176	1962.145	0.933	5.539	3491.957	0.923	6.481
	60	659.205	0.953	3.522	1441.212	0.951	3.715	2749.176	0.951	3.730
0.5	2	973.519	0.884	8.763	1900.333	0.886	8.502	3600	0.893	7.783
	15	866.206	0.965	0.588	1683.167	0.965	0.588	3367.439	0.962	0.867
	60	611.566	0.979	-0.950	1210.309	0.976	-0.599	2372.271	0.964	0.680
0.75	2	970.249	0.913	7.364	1947.587	0.911	7.512	3586.308	0.903	8.355
	15	830.585	0.987	-0.083	1691.112	0.986	-0.043	3309.959	0.973	1.341
	60	596.434	0.990	-0.436	1231.166	0.988	-0.236	2372.576	0.989	-0.331

Table 5.10: Sensitivity analysis for GRASP for (CD) using N60 instances.

	λ			
	0	0.25	0.5	0.75
`t`	218.11	210.16	186.95	184.47
`Value`	0.86	0.89	0.92	0.93
`RPD`	-0.06	0.09	0.05	0.06

Table 5.11: Sensitivity analysis for GRASP without random component for (CD) using N60 instances.

			`Value`	`t`	`RPD`
λ	0	k	6.16e-08	<2e-16	1.46e-08
		n_{iter_max}	0.390	<2e-16	0.355
	0.25	k	3.11e-05	<2e-16	1.46e-08
		n_{iter_max}	0.971	<2e-16	0.355
	0.5	k	0.0001	<2e-16	1.46e-08
		n_{iter_max}	0.983	<2e-16	0.355
	0.75	k	0.002	<2e-16	1.46e-08
		n_{iter_max}	0.920	<2e-16	0.355

Table 5.12: P-values of the ANOVA for the GRASP algorithm for (CD) with N60 instances.

For $\lambda \in \{0, 0.25, 0.5, 0.75\}$, we perform an ANOVA test to measure whether there is significant evidence between the impact of the different values of n_{iter_max} and k on `Value`, `t` and `RPD`. Table 5.12 presents the individual p-values. For both variables `Value` and `RPD`, there exists significant evidence to confirm that the different levels of parameter k affect them. We sketch the `RPD` for the different cases in Figure 5.7. For variable `t`, there exists significant evidence to confirm that the different levels of these two parameters affect it.

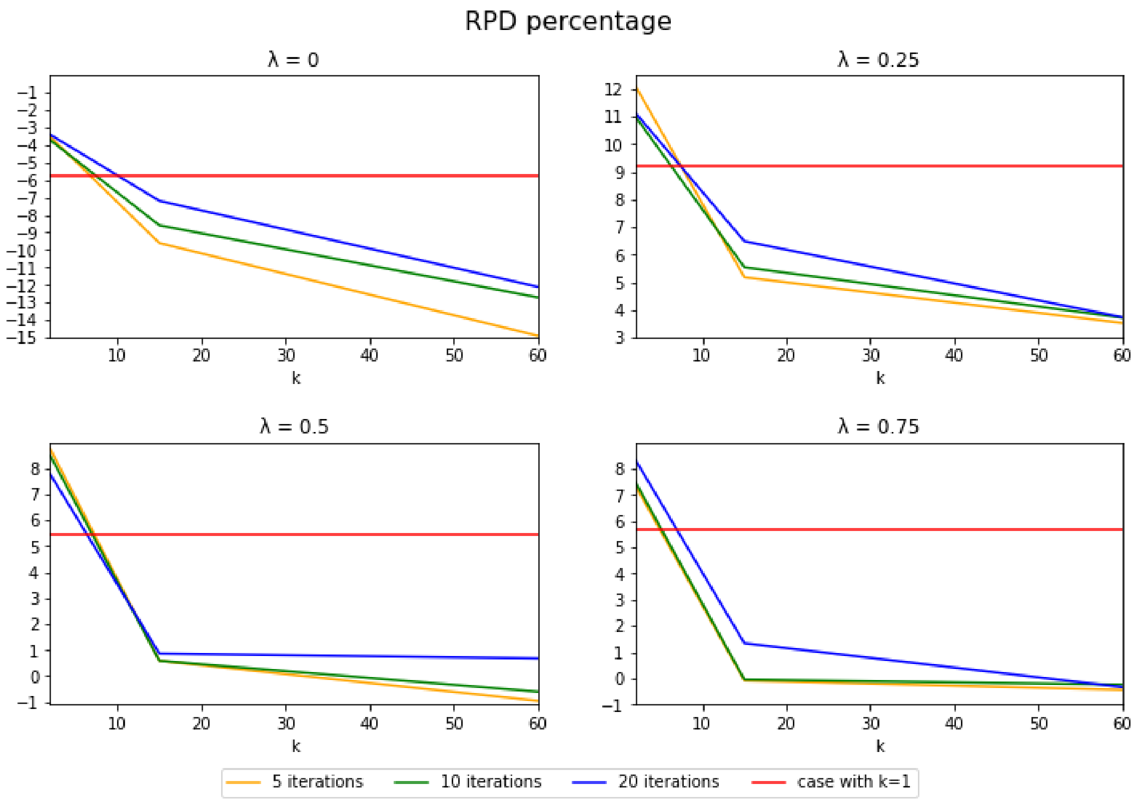


Figure 5.7: Sensitivity analysis for GRASP algorithm for (CD) for fixed values of λ using N60 instances.

By observing Table 5.10, in terms of `RPD`, GRASP always gets better results, except for the case $\lambda = 0$. Anyway, in all the cases, we observe that GRASP with values of parameter k close to 2 gets better solutions than in cases where it is large or it is fixed to 1. Besides, if parameter n_{iter_max} is set with large values, slightly better solutions are obtained in some cases, despite the fact that there is no significant evidence of it according to the ANOVA analysis.

The average CPU times for all of the combinations tested for GRASP are presented in Table 5.10. Combinations with large values of n_{iter_max} and small values of k take more time. Roughly speaking, for $\lambda \in \{0.25, 0.5, 0.75\}$, GRASP gets solutions which are on average 11.1%, 8.5% and 8.3%, respectively, (taking into account the best configuration, $k = 2$) better than the `BD_CW` solution in less than 1 hour. For $\lambda = 0$, in less than 1 hour GRASP gets solutions which are on average 3% (taking into account the best configuration) different from the `BD_CW` solution found.

We found that GRASP has a poor performance for $\lambda \geq 1$. This may be since, in the first iterations of Algorithm 10, the randomly selected edges do not minimize the objective function of the center problem. In other words, adding individual *good* edges does not necessarily minimize the maximum shortest path in the network.

5.6 Conclusions

In this chapter, we have introduced and studied for the first time the λ-cent-dian and the generalized-center problems for Network Design when the demand is given by pairs of points instead of single points. These problems minimize a linear combination of the maximum and average traveled distances. While the λ-cent-dian problem, with $\lambda \in [0, 1]$, minimizes a convex combination of both objectives, the generalized-center minimizes the difference between them. We have studied these concepts under two versions of Pareto-optimality: the first considers the shortest paths of each O/D pair, and the second one deals with both objective functions. With respect to the second one, we have checked that for generating the whole set of Pareto-optimal solutions the new concept of maximum λ-cent-dian has to be introduced.

The generalized-center solutions are often not reasonable in terms of efficiency since the median value is increased artificially. Hence, we add to this problem an efficiency constraint that ensures that the median value is not too far from the median function value of the median network. This constrained generalized-center can be seen as analogous to performing a lexicographic optimization of the median and generalized-center objectives.

Given the hardness of these problems, for the case $\lambda \in [0, 1]$ we have studied and formulated both: a branch-and-Benders-cut method; and a GRASP to find good solutions in reasonable computational time. Our computational results show that our branch-and-Benders-cut approach for (CD) is: i) competitive against the one proposed by `CPLEX` for instances with more than 60 nodes, and ii) non-dominated by the GRASP procedure for instances with 60 nodes.

Finally, we have evaluated the λ-cent-dian, with $\lambda \in [0, 1]$, and generalized-center solutions using some inequality measures. In this situation, taking into account the efficiency constraint, we have verified that the generalized-center solution is not always dominated.

Chapter 6

Conclusions and future research lines

This thesis deals with Network Design problems with covering and λ-cent-dian objectives. All of them have been approached from the Mathematical Programming point of view. We have proposed MILP formulations which have been analyzed and properties obtained. After that, we have developed some techniques to solve them.

Next, we briefly summarize the major achievements of each chapter and discuss possible further research lines.

In Chapter 2, two variants of the Covering Network Design Problem have been studied: Maximal-Covering Network Design problem and Partial-Covering Network Design problem. The first maximizes the demand covered under a budget constraint. The latter minimizes the total constructing cost subject to a lower bound on the demand covered. For solving them, mixed-integer linear programming formulations have been proposed, which are stronger than the existing ones. Besides, with the aim of maximizing the coverage and minimizing the cost, we have proposed a sequential procedure based on the formulations studied. Moreover, some polyhedral properties of these formulations are provided, which are useful for the algorithmic techniques used to solve the problems. We have developed exact methods based on Benders decomposition. We also discuss some preprocessing procedures to scale up the instances solved. With these preprocessing techniques we obtain information about the instances, which is useful to derive a better algorithmic performance. Our computational results show that the techniques developed in this chapter are competitive against the ones existing in the literature since they allow obtaining better solutions in less time. Further research on this topic is focused on the synergy of sophisticated heuristics to find good feasible solutions and decomposition methods, such as the ones presented in this chapter, to get better bounds and close the optimality gap. This further research has been started for one of the two problems presented. In Chapter 4, we have designed some sophisticated heuristics. Finally, we remark that objectives of (MC) and (PC) can be included in a bi-criteria optimization model. An interesting extension is to exploit the decomposition methods described in this chapter to the multiobjective setting.

Motivated by real-life applications, in Chapter 3 we have presented a formulation which locates a rapid transit line and modifies the old route of a slow transit line simultaneously with the purpose of maximizing the demand covered. We have focused on the case study in which a single transfer station is allowed and the end-points of the lines do not coincide. In addition, it is shown in an example with the `Seville` network instance that the design of both lines in an integrated way obtains greater coverage than that of sequentially designing (first the rapid transit line is designed and then the slow line is re-located). To solve this problem, we have developed a branch-and-Benders-cut scheme specifically for the formulation discussed. Using the `Seville` network instance, we have obtained that the Benders approach that uses the ideas of Conforti and Wolsey (2019) is competitive against the standard decomposition of Benders (Subsection 3.3.1), the Benders procedure existing in `CPLEX` and that of directly using `CPLEX`. Possible further research will go in the direction of considering the design of more than one rapid transit line as well as the re-location of more than one slow transit line in an integrated way. It will be interesting to explore the solution depending on the presence or not of cycles between the set of lines and the maximum number of transfer stations to be considered. Furthermore, it could be adequate to separate the real-case studies (large metropolitan areas or cities) into sectors to take into account and consider their features, such as the topology. Finally, developing a Benders decomposition to generate facet-defining Benders cuts for this much more complex case could be also efficient and advantageous.

In Chapter 4, two well-known metaheuristic techniques have been adapted to the Maximal-Covering Network Design problem studied in Chapter 2. One of them is a Simulated Annealing algorithm and the other is an Adaptive Large Neighborhood Search procedure. The main difference between them is that the second one works with a larger neighborhood. An extensive sensitivity analysis has been carried out for both using randomly generated medium size instances. In addition, we have added to this computational study the Genetic Algorithm provided in Perea et al. (2020). The goal has been to observe the behavior of the parameters and to find the best parameter settings in each case. The ANOVA analysis done for each of the metaheuristic showed that not all the parameters are individually significant to the quality of the solutions obtained, according to the set of values considered for such parameters. Another important conclusion is that the solutions obtained by these metaheuristics could be better than those obtained by other procedures based on exact methods, as the `BD_CW` implementation developed, if the instance in question is considerably large. As a final task, we have added a computational comparison between them using such best configurations for a large instance. As further research, it would be interesting to develop some ideas related to matheuristic routines, which take advantage of both, exact and metaheuristic procedures. The difficult part is solved with some heuristic approach and the easiest one is solved using exact methods.

In Chapter 5, we have introduced and studied for the first time the λ-cent-dian and the generalized-center problems for Network Design when the demand is given by pairs of points instead of single points. These problems minimize a linear combination of the maximum and average traveled distances. While the λ-cent-dian problem, with $\lambda \in [0, 1]$, minimizes a convex combination of both objectives, the generalized-center minimizes the difference between them. We have studied these concepts under two versions of Pareto-optimality: the first considers the shortest paths of each O/D pair, and the second one deals with both objective functions. With respect to the second one, we have checked that for generating the whole set of Pareto-optimal solutions the new concept of maximum λ-cent-dian has to be introduced. On the other hand, the generalized-center solutions are often not reasonable in terms of efficiency since the median value is increased artificially. Hence, we add to this problem an efficiency constraint that ensures that the median value is not too far from the median function value of the median network. This constrained generalized-center can be seen as analogous to performing a lexicographic optimization of the median and generalized-center objectives. Given the hardness of these problems, for the case $\lambda \in [0,1]$ we have studied and formulated both: a branch-and-Benders-cut method, and a GRASP to find good solutions in reasonable computational time. Our computational results show that our branch-and-Benders-cut approach for the λ-cent-dian formulation, with $\lambda \in [0,1]$, is: i) competitive against the one proposed by `CPLEX` for instances with more than 60 nodes, and ii) non-dominated by the GRASP procedure for instances with 60 nodes. Finally, we have evaluated the λ-cent-dian, with $\lambda \in [0,1]$, and generalized-center solutions using some inequality measures. In this situation, taking into account the efficiency constraint, we have verified that the generalized-center solution is not always dominated. Further research of this work could be adding to the formulations a coverage constraint to ensure that at least a certain percentage of demand is covered by the prospective network to be designed. Besides, it would be interesting to consider bi-objective functions composed by the center or the median together with a coverage or cost function.

Finally, note that a common characteristic in all the problems studied is that of the existence of an alternative mode/network to satisfy the demand flows. This situation has been modeled using binary variables. That is, the flow cannot be shared between both modes, which does not represent reality. We emphasize that the goal has been to develop good resolution methods, and then use the achievements to solve more realistic models. Therefore, a future line of research, and common in all chapters, is to model the flows using approximated random utility models, e.g. piecewise linear approximations of logit functions.

Chapter 7

Conclusiones y líneas de investigación futuras

En esta tesis se estudian problemas de Diseño de Redes con objetivos relacionados con el cubrimiento de demanda y con las nociones de centro y mediana (λ-centdian). Todos ellos han sido enfocados desde el punto de vista de la Programación Matemática. Hemos propuesto formulaciones lineales entero-mixtas, las cuáles han sido analizadas y algunas propiedades obtenidas. Después de esto, hemos desarrollado algunas técnicas para resolver tales problemas.

A continuación, explicamos brevemente los logros más relevantes de cada capítulo y exponemos posibles líneas de investigación futuras.

En el Capítulo 2 se estudian dos variantes del Problema de Cubrimiento para el Diseño de Redes: el problema de Cubrimiento-Máximo para el Diseño de Redes y el problema de Cubrimiento-Parcial para el Diseño de Redes. El primero maximiza la demanda a cubrir bajo una restricción de presupuesto. El segundo minimiza el coste total de construcción sujeto a una restricción que acota inferiormente la demanda a cubrir. Para resolverlos, se han propuesto formulaciones de programación lineal entero-mixtas, las cuáles son más fuertes que las ya existentes en la literatura. Además, con el objetivo de maximizar el cubrimiento y minimizar el coste, hemos propuesto un procedimiento secuencial basado en las formulaciones estudiadas. También, hemos proporcionado algunas propiedades poliédricas de estas formulaciones, útiles para las técnicas algorítmicas utilizadas para resolver los problemas. Hemos desarrollado métodos exactos basados en la descomposición de Benders. También hemos propuesto algunos procedimientos de preprocesamiento para escalar las instancias resueltas. Con estas técnicas de preprocesamiento obtenemos información sobre las instancias, la cuál es útil para conseguir un mejor comportamiento algorítmico. Nuestros resultados computacionales muestran que las técnicas desarrolladas en este capítulo son competitivas con las ya existentes en la literatura, ya que con ellas se obtienen mejores soluciones con un menor tiempo de cómputo. Investigaciones futuras relacionadas con este tema pueden centrarse en la sinergia de heurísticas sofisticadas para encontrar buenas soluciones y métodos de descomposición, tales como los presentados en este capítulo, para conseguir mejores cotas y acercarnos más a la solución óptima. El trabajo desarrollado en el Capítulo 4 puede ser considerado como el comienzo de esta propuesta de trabajo futura para uno de los dos problemas presentados. Esto es, en el Capítulo 4, hemos diseñado algunas heurísticas más sofisticadas para el problema de Cubrimiento-Máximo. Finalmente, remarcamos que los objetivos de los problemas de Cubrimiento-Máximo y Cubrimiento-Parcial pueden ser incluidos en un modelo de optimización con una función bi-criterio. Una extensión interesante puede ser la de aplicar los métodos de descomposición descritos en este capítulo al caso de la optimización multiobjetivo utilizando tales funciones.

Motivados por aplicaciones a la vida real, en el Capítulo 3 hemos presentado una formulación que localiza una línea de tránsito rápido y modifica una línea de tránsito lento de manera simultánea, con el objetivo de maximizar la demanda a cubrir. Nos hemos centrado en el caso de estudio en el que solo se permite construir una estación de transbordo y en el que los puntos iniciales y finales de las líneas no son coincidentes. Además, utilizando la ciudad de Sevilla como ejemplo, se ha mostrado que con el diseño de manera integrada se consigue un mayor cubrimiento que con el diseño secuencial (primero localizar la red de tránsito rápido y luego relocalizar la red de tránsito lento). Para resolver este problema, hemos desarrollado, específicamente para la formulación discutida, un esquema de Ramificación-y-Corte que genera cortes de Benders. Utilizando la ciudad de Sevilla como ejemplo, hemos obtenido que el enfoque propuesto que utiliza las ideas de Conforti and Wolsey (2019) es competitivo contra la descomposición estándar de Benders (Subsección 3.3.1), contra el procedimiento de Benders existente en `CPLEX` y contra el uso directo de `CPLEX`. Posibles líneas de investigación futuras pueden ir en la dirección de considerar el diseño de más de una línea de tránsito rápido, así como la relocalización de más de una línea de tránsito lento de manera simultánea. Sería interesante analizar la solución conforme a la presencia o no de ciclos entre el conjunto de líneas y conforme al máximo número de estaciones de transbordo a considerar. Además, podría ser adecuado separar los casos de estudio reales (ciudades o áreas metropolitanas de gran tamaño) en sectores, para tener en cuenta características particulares, tales como la topología de la zona. Finalmente, también podría ser ventajoso y eficiente desarrollar una descomposición de Benders para generar cortes que definan facetas para este caso de estudio mucho más complejo.

En el Capítulo 4, dos técnicas metaheurísticas muy conocidas han sido adaptadas para el problema de Cubrimiento-Máximo en el Diseño de Redes estudiado en el Capítulo 2. Una de ellas es el algoritmo del Recocido Simulado y la otra es un procedimiento de Búsqueda Adaptativa considerando vecindarios de gran tamaño. La principal diferencia entre ellas es que la segunda trabaja con un vecindario más amplio. Par ambos algoritmos se ha hecho un análisis de sensibilidad extenso, utilizando instancias de tamaño medio generadas aleatoriamente. Además, hemos añadido a este estudio computacional el algoritmo genético propuesto en Perea et al. (2020). El objetivo ha sido observar el comportamiento de los parámetros y encontrar la mejor configuración de ellos en cada caso. El análisis ANOVA realizado para cada una de las metaheurísticas dio como resultado que no todos los parámetros son individualmente significativos para la calidad de las soluciones obtenidas, según el conjunto de valores considerados para tales parámetros. Otra conclusión importante es que las soluciones obtenidas por estas metaheurísticas podrían ser mejores que las obtenidas por otros procedimientos basados en métodos exactos, como la implementación desarrollada de `BD_CW`, si la instancia en cuestión es considerablemente grande. Como tarea final, hemos agregado una comparación computacional entre ellos utilizando las mejores configuraciones para una instancia grande. Como investigación futura, sería interesante desarrollar algunas ideas relacionadas con procedimientos mateheurísticos, que toman ventaja tanto de los procedimientos exactos como de los metaheurísticos. La parte difícil se resuelve con algún enfoque heurístico y la más fácil se resuelve usando métodos exactos.

En el Capítulo 5 presentamos y estudiamos por primera vez los problemas de λ-cent-dian y del centro-generalizado para el diseño de redes cuando la demanda está dada por pares de puntos en lugar de puntos individuales. Estos problemas minimizan una combinación lineal de la distancia máxima (funciín centro) y la distancia media (función mediana) recorrida. Mientras que el problema λ-cent-dian, con $\lambda \in [0, 1]$, minimiza una combinación convexa de ambos objetivos, el centro-generalizado minimiza la diferencia entre ellos. Hemos estudiado estos conceptos bajo dos versiones de la optimización de Pareto: la primera considera los caminos más cortos de cada par O/D, y la segunda se ocupa de ambas funciones objetivo. Con respecto a la segunda, hemos comprobado que para generar todo el conjunto de soluciones Pareto-óptimas se ha de introducir el nuevo concepto de λ-cent-dian máximo. Por otro lado, las soluciones del centro-generalizado a menudo no son razonables en términos de eficiencia, ya que la distancia media se incrementa artificialmente. Por lo tanto, agregamos a este problema una restricción de eficiencia que asegura que el valor de la mediana no se aleje demasiado del valor de la función mediana en la red mediana. Este problema de centro-generalizado restringido puede verse como análogo al problema de realizar una optimización lexicográfica tomando los objetivos de la mediana y del centro-generalizado. Dada la dureza de estos problemas, para el caso $\lambda \in [0, 1]$ hemos estudiado y formulado ambos: un método de Ramificación-y-Corte que utiliza la teoría de Benders, y un GRASP para encontrar buenas soluciones en un tiempo computacional razonable. Nuestros resultados computacionales muestran que nuestra propuesta de algoritmo de Ramificación-y-Corte con la teoría de Benders es: i) competitivo frente al existente en `CPLEX` para instancias con más de 60 nodos, y ii) no está dominado por el procedimiento GRASP para instancias con 60 nodos. Finalmente, hemos evaluado las soluciones del λ-cent-dian, con $\lambda \in [0, 1]$, y del centro-generalizado usando algunas medidas de desigualdad. En esta situación, teniendo en cuenta la restricción de eficiencia, hemos comprobado que la solución de centro-generalizado no siempre está dominada. La investigación futura para este trabajo podría agregar a las formulaciones una restricción de cobertura para garantizar que al menos un cierto porcentaje de la demanda sea cubierto por la red prospectiva a diseñar. Además, sería interesante considerar funciones bi-objetivo compuestas por el centro o la mediana junto con una función de cobertura o costo.

Finalmente, nótese que una característica común en todos los problemas estudiados es la existencia de un modo/red alternativo para satisfacer los flujos de demanda.

Esta situación ha sido modelada utilizando variables binarias. Es decir, el flujo no se puede compartir entre ambos modos, lo cuál no representa la realidad. Hacemos hincapié en que el objetivo ha sido desarrollar buenos métodos de resolución, para luego utilizar los logros obtenidos para resolver modelos más realistas. Por tanto, una futura línea de investigación, y común en todos los capítulos, sería la de modelar los flujos utilizando modelos aproximados de utilidad aleatoria, por ejemplo, aproximaciones lineales a trozos de funciones logit.

Appendix A

Pseudo-code for initial feasible solutions for (MC) and (PC)

In this appendix we provide the pseudo-codes to determine an initial feasible solution for the (MC) and (PC) formulations described in Section 2.2.4. We denote by N_s and E_s the sets of nodes and edges to design at the end of each algorithm. Besides, the set W_s refers to the mode choice variables set to 1 once the algorithm is done.

Algorithm 12: Initial Feasible Solution for (MC)

```
**Initialization:** Set $N_s = \emptyset$, $E_s = \emptyset$ and $W_s = \emptyset$ and $IC = 0$.
Compute ratio $r^w = \frac{g^w}{C(\text{Path}^w)}$:
**for** $w \in W$ *in decreasing order of* $r^w$ **do**
    $\bar{C} = C(\text{Path}^w) - \sum_{e \in E_s \cap \widetilde{E}^w} c_e - \sum_{i \in N_s \cap \widetilde{N}^w} b_i$.
    **if** $IC + \bar{C} \leq C_{max}$ **then**
        $W_s \leftarrow W_s \cup \{w\}$
        $E_s \leftarrow E_s \cup \widetilde{E}^w$
        $N_s \leftarrow N_s \cup \widetilde{N}^w$
        $IC \leftarrow IC + \bar{C}$
    **end**
**end**
$x_e = 1$ for $e \in E_s$, 0 otherwise.
$y_i = 1$ for $i \in N_s$, 0 otherwise.
$z^w = 1$ for $w \in W_s$, 0 otherwise.
**Return:** $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$
```

Algorithm 13: Initial Feasible Solution for (PC)

```
**Initialization:** Set $\bar{W}_s = W$ and $G_s = G_{total}$.
Compute ratio $r^w = \frac{g^w}{C(\text{Path}^w)}$:
**for** $w \in W$ *in decreasing order of* $r^w$ **do**
    **if** $G_s - g^w \geq \beta \, G_{total}$ **then**
        $W_s \leftarrow W_s \setminus \{w\}$
        $G_s \leftarrow G_s - g^w$
    **end**
**end**
$x_e = 1$ if $e \in \bigcup_{w \in W_s} \widetilde{E}^w$, 0 otherwise.
$y_i = 1$ if $i \in \bigcup_{w \in W_s} \widetilde{N}^w$, 0 otherwise.
$z^w = 1$ for $w \in W_s$, 0 otherwise.
**Return:** $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$
```

Appendix B

Results for Sioux Falls city network for (MC) and (PC)

These figures show some results of a sensitivity analysis done for the `BD_CW` implementation developed in Chapter 2 using the benchmark instance of `Sioux Falls`.

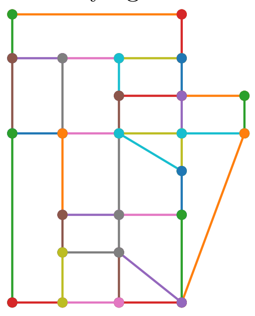
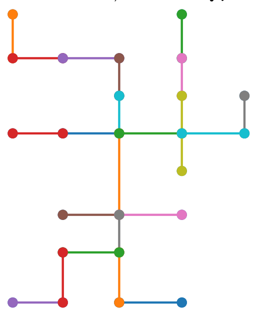
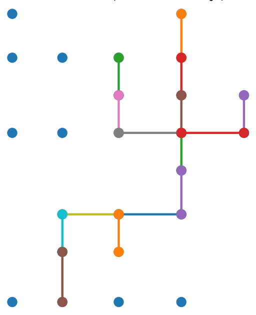
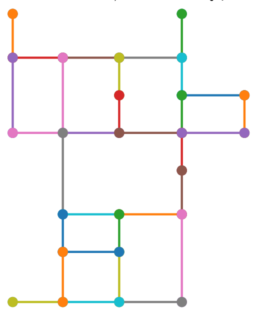
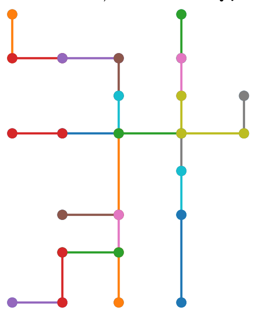
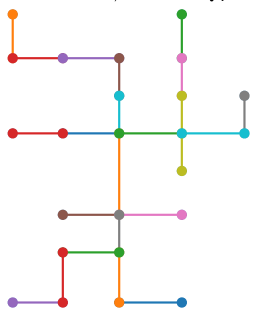
Underlying Network  $C_{total} = 4171$, $G_{total} = 84437$	$\alpha = 0.5$, $u = 2\,d_{\mathcal{N}}$  `t` $= 22.85$, `cuts` $= 3496$ `cost` $= 2070$, `v(ILP)` $= 75488$
$\alpha = 0.3$, $u = 2\,d_{\mathcal{N}}$  `t`$= 458.84$, `cuts` $= 3056$ `cost` $= 1237$, `v(ILP)` $= 35039$	$\alpha = 0.7$, $u = 2\,d_{\mathcal{N}}$  `t` $= 2.73$, `cuts` $= 801$ `cost` $= 2870$, `v(ILP)` $= 82699$
$\alpha = 0.5$, $u = 1.5\,d_{\mathcal{N}}$  `t`$= 60.31$, `cuts` $= 3460$ `cost` $= 2080$, `v(ILP)` $= 68227$	$\alpha = 0.5$, $u = 3\,d_{\mathcal{N}}$  `t` $= 14.17$, `cuts` $= 2641$ `cost` $= 2070$, `v(ILP)` $= 75488$

Figure B.1: Sensitivity analysis for (MC) using `Sioux Falls` instance.

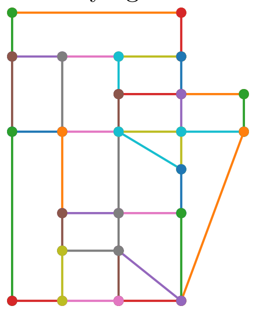
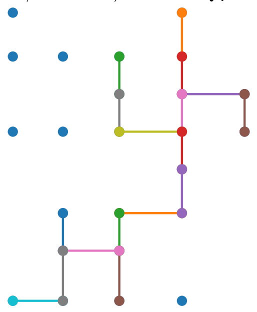
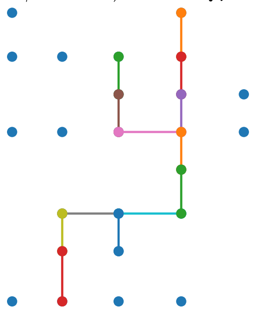
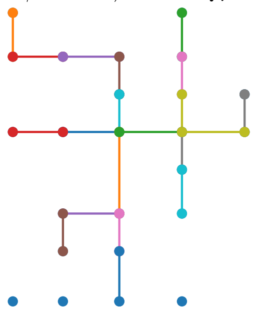
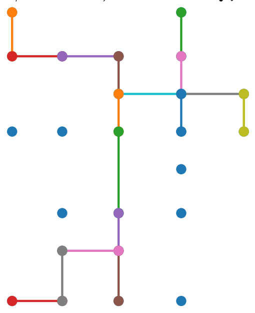
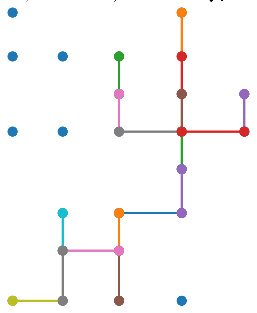
Underlying Network  $C_{total} = 4171$, $G_{total} = 84437$	$\beta = 0.5$, $u = 2\,d_{\mathcal{N}}$  `t` $= 429.85$, `cuts` $= 3306$ $G_{cov} = 44112$, `v(ILP)` $= 1411$
$\beta = 0.3$, $u = 2\,d_{\mathcal{N}}$  `t` $= 925.68$, `cuts` $= 2783$ $G_{cov} = 24588$, `v(ILP)` $= 1058$	$\beta = 0.7$, $u = 2\,d_{\mathcal{N}}$  `t` $= 136.06$, `cuts` $= 3674$ $G_{cov} = 60276$, `v(ILP)` $= 1726$
$\beta = 0.5$, $u = 1.5\,d_{\mathcal{N}}$  `t` $= 1471.84$, `cuts` $= 3793$ $G_{cov} = 43599$, `v(ILP)` $= 1491$	$\beta = 0.5$, $u = 3\,d_{\mathcal{N}}$  `t` $= 1149.26$, `cuts` $= 3128$ $G_{cov} = 42331$, `v(ILP)` $= 1411$

Figure B.2: Sensitivity analysis for (PC) using `Sioux Falls` instance.

Appendix C

Results for Sioux Falls city networks for (CD)

		CPLEX		Auto_BD			BD_CW		
α	λ	t	gap	t	gap	cuts	t	gap	cuts
0.15	0	**3.7812**	0	4.4219	0	699	10.25	0	2716
	0.25	**4.75**	0	5.1094	0	835	17.9688	0	3391
	0.5	**2.6562**	0	2.875	0	754	6.0781	0	1306
	0.75	**0.78**	0	2.56	0	712	2.54	0	1206
	1	**0.26**	0	0.96	0	132	0.78	0	630
0.25	0	**151.51**	0	259.25	0	1648	799.72	0	9201
	0.25	**196.20**	0	206.09	0	2517	578.84	0	8358
	0.5	**175.90**	0	327.54	0	2925	507.54	0	12261
	0.75	**625.51**	0	725.87	0	2474	734.08	0	10275
	1	**4.11**	0	6.75	0	189	12.20	0	5757
0.4	0	388.65	0	**302.15**	0	3295	1192.15	0	7095
	0.25	168.73	0	**81.29**	0	3071	85.18	0	6273
	0.5	76.09	0	**53.61**	0	2247	108.26	0	5065
	0.75	125.28	0	**70.44**	0	2571	114.90	0	5703
	1	15.45	0	**8.89**	0	315	23.41	0	4642

Table C.1: Sensitivity analysis of BD_CW implementation for (CD) formulation with Sioux Falls instance.

List of Figures

List of Tables

Notation

Set	Model	Description
$\mathcal{N}$	all	Potential graph to consider.
$\mathcal{S}$	chapter 5	Represent a subgraph of $\mathcal{N}$.
$\mathcal{R}$, $\mathcal{S}$	chapter 3	Rapid and slow transit lines to design.
N	all	The set of nodes of graph $\mathcal{N}$.
E	all	The set of edges of graph $\mathcal{N}$.
$N_{\mathcal{R}}$, $N_{\mathcal{S}}$	chapter 3	The set of nodes of line graphs $\mathcal{R}$ and $\mathcal{S}$.
N_{trans}	chapter 3	Set of transfer stations.
$E_{\mathcal{S}}$	chapter 5	The set of edges of subgraph $\mathcal{S}$.
$E_{\mathcal{R}}$, $E_{\mathcal{S}}$	chapter 3	The set of edges of line graphs $\mathcal{R}$ and $\mathcal{S}$.
A	all	The set of arcs of graph $\mathcal{N}$.
$A_{\mathcal{S}}$	chapter 5	The set of arcs of subgraph $\mathcal{S}$.
$A_{\mathcal{R}}$, $A_{\mathcal{S}}$	chapter 3	The set of arcs of line graphs $\mathcal{R}$ and $\mathcal{S}$.
W	all	The demand set of origin/destination pairs.
$\mathcal{N}^w$	chapters 2,3,5	Feasible subgraph of the origin/destination pair w.
N^w	chapters 2,3,5	The set of nodes of subgraph $\mathcal{N}^w$.
E^w	chapters 2,3,5	The set of edges of subgraph $\mathcal{N}^w$.
$O_{\mathcal{R}}$, $O_{\mathcal{S}}$	chapter 3	The set of possible starting points of lines $\mathcal{R}$ and $\mathcal{S}$, respectively.
$D_{\mathcal{R}}$, $D_{\mathcal{S}}$	chapter 3	The set of possible end points of lines $\mathcal{R}$ and $\mathcal{S}$, respectively.
$\delta(i)$	all	The set of edges incident to node i. In chapter 3, it refers to the set of edges of $E_{\mathcal{R}}$ incident to node i.
$\delta^w(i)$	all	The set of edges incident to node i in the subgraph $\mathcal{N}^w$. In chapter 3, it refers to the set of edges of $E_{\mathcal{R}}$ incident to node i in graph $\mathcal{N}^w$.
$\delta^w_-(i)$	all	The set of arcs going in of node i in the subgraph $\mathcal{N}^w$. In chapter 5, it refers to the set of arcs of $A_{\mathcal{R}}$ going in of node $i \in N^w$.
$\delta^w_+(i)$	all	The set of arcs going out of node i in the subgraph $\mathcal{N}^w$. In chapter 5, it refers to The set of arcs of $A_{\mathcal{S}}$ going out of node $i \in N^w$.
$\vartheta(i)$	chapter 3	The set of edges of $E_{\mathcal{S}}$ incident to node i.
$\vartheta^w(i)$	chapter 3	The set of edges of $E_{\mathcal{S}}$ incident to node i in the graph N^w.
$\vartheta^w_-(i)$	chapter 3	The set of arcs of $A_{\mathcal{S}}$ going in of node $i \in N^w$.
$\vartheta^w_+(i)$	chapter 3	The set of arcs of $A_{\mathcal{S}}$ going out of node $i \in N^w$.
$\boldsymbol{\mathcal{PO}}^\alpha$	chapter 5	The set of Pareto-optimal solution networks with respect to the distance of the shortest paths.
$\boldsymbol{\mathcal{PO}}^\alpha_2$	chapter 5	The set of Pareto-optimal solution networks with respect to the bicriteria center/median problem.

Table 1: List of notation for the sets of elements used in the problems studied.

Element	Model	Description
i, j, k	all	Nodes
$e = \{i, j\}$	all	Edge
$a = (a^s, a^t)$	all	Arc going out of node a^s and going in a^t.
$w = (w^s, w^t)$	all	Origin-destination pair.

Table 2: List of notation for the elements used in the problems studied.

Parameters	Model	Description
b_i	chapters 2,4,5	Construction cost of node i.
c_e	chapters 2,4,5	Construction cost of edge e.
C_{total}	chapters 2,4,5	Construction cost of the whole potential network $\mathcal{N}$.
d_e, d_a	all	Weight assigned to an edge, arc, respectively.
g^w	all	Demand associated to the origin/destination pair w.
G_{total}	chapters 2, 5	Total demand.
u^w	all	Length associated to an unknown predefined path in an alternative existing network.
$d_{\mathcal{S}}(w)$	chapter 5	Length of the shortest path from w^s to w^t in the network $\mathcal{S}$.
$\ell_{\mathcal{S}}(w)$	chapter 5	Minimum value between $d_{\mathcal{S}}(w)$ and u^w for a given dmand pair w.
α, β	chapters 2,4,5	Proportion of C_{total} to consider.
β	chapter 2	Proportion of G_{total} to consider.
$E^{max}_{\mathcal{R}}$, $E^{max}_{\mathcal{S}}$	chapter 3	Maximum number of edges to be constructed in $\mathcal{R}$ and $\mathcal{S}$.
$E^{id}_{\mathcal{S}}$	chapter 3	Minimum number of edges to be coincident with the old design of $\mathcal{S}$.
$v^{\mathcal{S}}_e$	chapter 3	Current path of the slow line $\mathcal{S}$.
$t^{\mathcal{R}}_a$, $t^{\mathcal{S}}_a$	chapter 3	Cost of traversing arc a in the rapid and slow mode, respectively.
$t^{\mathcal{RS}}_k$, $t^{\mathcal{SR}}_k$	chapter 3	Transfer costs at station k from $\mathcal{R}$ to $\mathcal{S}$ and from $\mathcal{S}$ to $\mathcal{R}$.
$t^{\mathcal{R}}_{stop}$, $t^{\mathcal{S}}_{stop}$	chapter 3	Dwell time costs.
t_{wait}	chapter 3	Waiting time at stations.

Table 3: List of notation for the parameters used in the problems studied.

References

Ahuja, R. K., Ergun, Ö., Orlin, J. B., and Punnen, A. P. (2002). A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1-3):75–102.

Averbakh, I. and Berman, O. (1999). Algorithms for path medi-centers of a tree. *Computers & Operations Research*, 26(14):1395–1409.

Balakrishnan, A., Magnanti, T. L., Shulman, A., and Wong, R. T. (1991). Models for planning capacity expansion in local access telecommunication networks. *Annals of Operations Research*, 33(4):237–284.

Balakrishnan, A., Magnanti, T. L., and Wong, R. T. (1989). A dual-ascent procedure for large-scale uncapacitated network design. *Operations Research*, 37(5):716–740.

Balas, E. and Perregaard, M. (2002). Lift-and-project for mixed 0–1 programming: recent progress. *Discrete Applied Mathematics*, 123(1-3):129–154.

Balas, E. and Perregaard, M. (2003). A precise correspondence between lift-and-project cuts, simple disjunctive cuts, and mixed integer gomory cuts for 0-1 programming. *Mathematical Programming*, 94(2-3):221–245.

Barahona, F. (1996). Network design using cut inequalities. *SIAM Journal on Optimization*, 6(3):823–837.

Bärmann, A. (2016). *Solving network design problems via decomposition, aggregation and approximation*. Springer.

Ben-Ameur, W. and Neto, J. (2007). Acceleration of cutting-plane and column generation algorithms: Applications to network design. *Networks: An International Journal*, 49(1):3–17.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252.

Berge, C. (1957). Two theorems in graph theory. *Proceedings of the National Academy of Sciences of the United States of America*, 43(9):842–844.

Bern, M. and Bienstock, D. (1991). Polynomially solvable special cases of the steiner problem in planar networks. *Annals of Operations Research*, 33(6):403–418.

Botton, Q., Fortz, B., Gouveia, L., and Poss, M. (2013). Benders decomposition for the hop-constrained survivable network design problem. *INFORMS journal on computing*, 25(1):13–26.

Bucarey, V., Fortz, B., González-Blanco, N., Labbé, M., and Mesa, J. A. (2022). Benders decomposition for network design covering problems. *Computers & Operations Research*, 137:105417.

Canca, D., De-Los-Santos, A., Laporte, G., and Mesa, J. A. (2016). The railway network design, line planning and capacity problem: An adaptive large neighborhood search metaheuristic. In *Advanced Concepts, Methodologies and Technologies for Transportation and Logistics*, pages 198–219. Springer.

Canca, D., De-Los-Santos, A., Laporte, G., and Mesa, J. A. (2017). An adaptive neighborhood search metaheuristic for the integrated railway rapid transit network design and line planning problem. *Computers & Operations Research*, 78:1–14.

Canca, D., De-Los-Santos, A., Laporte, G., and Mesa, J. A. (2019). Integrated railway rapid transit network design and line planning problem with maximum profit. *Transportation Research Part E: Logistics and Transportation Review*, 127:1–30.

Cascetta, E. (2009). *Transportation systems analysis: models and applications*, volume 29. Springer Science & Business Media, New York.

Černỳ, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51.

Chou, H., Premkumar, G., and Chu, C.-H. (2001). Genetic algorithms for communications network design-an empirical study of the factors that influence performance. *IEEE Transactions on Evolutionary Computation*, 5(3):236–249.

Chouman, M. and Crainic, T. G. (2015). Cutting-plane matheuristic for service network design with design-balanced requirements. *Transportation Science*, 49(1):99–113.

Church, R. and ReVelle, C. (1974). The maximal covering location problem. In *Papers of the Regional Science Association*, volume 32, pages 101–118. Springer-Verlag.

Coelho, L. C., Cordeau, J.-F., and Laporte, G. (2012). The inventory-routing problem with transshipment. *Computers & Operations Research*, 39(11):2537–2548.

Conforti, M. and Wolsey, L. A. (2019). “Facet” separation with one linear program. *Mathematical Programming*, 178(1):361–380.

Contreras, I. (2021). Hub network design. In Crainic, T. G., Gendreau, M., and Gendron, B., eds., *Network Design with Applications to Transportation and Logistics*, pages 567–598. Springer.

Contreras, I. and Fernández, E. (2012). General network design: A unified view of combined location and network design problems. *European Journal of Operational Research*, 219(3):680–697.

Cordeau, J.-F., Furini, F., and Ljubić, I. (2019). Benders decomposition for very large scale partial set covering and maximal covering location problems. *European Journal of Operational Research*, 275(3):882–896.

Costa, A. M., Cordeau, J.-F., and Gendron, B. (2009). Benders, metric and cutset inequalities for multicommodity capacitated network design. *Computational Optimization and Applications*, 42(3):371–392.

Crainic, T. G., Gendreau, M., and Gendron, B. (2021). *Network Design with Applications to Transportation and Logistics*. Springer, Canada.

Dempe, S. and Zemkoho, A. (2020). Bilevel optimization. In *Springer optimization and its applications. Vol. 161*. Springer.

Deng, L., Gao, W., Zhou, W., and Lai, T. (2013). Optimal design of feeder-bus network related to urban rail line based on transfer system. *Procedia-Social and Behavioral Sciences*, 96:2383–2394.

Desrochers, M. (1986). *An algorithm for the shortest path problem with resource constraints*, volume 421. Université de Montréal, Centre de recherche sur les transports, Canada.

Eglese, R. W. (1990). Simulated annealing: a tool for operational research. *European Journal of Operational Research*, 46(3):271–281.

Fan, W. and Machemehl, R. B. (2006). Using a simulated annealing algorithm to solve the transit route network design problem. *Journal of Transportation Engineering*, 132(2):122–132.

Fathollahi-Fard, A. M., Hajiaghaei-Keshteli, M., Tian, G., and Li, Z. (2020). An adaptive lagrangian relaxation-based algorithm for a coordinated water supply and wastewater collection network design problem. *Information Sciences*, 512:1335–1359.

Feillet, D., Dejax, P., Gendreau, M., and Gueguen, C. (2004). An exact algorithm for the elementary shortest path problem with resource constraints: Application to some vehicle routing problems. *Networks: An International Journal*, 44(3):216–229.

Fischetti, M., Salvagnin, D., and Zanette, A. (2010). A note on the selection of Benders' cuts. *Mathematical Programming*, 124(1-2):175–182.

Forsgren, A. and Prytz, M. (2008). *Telecommunications Networks Design*, pages 269–290. Springer Science & Business Media, U S A.

Fortz, B., Gouveia, L., and Moura, P. (2021). A comparison of node-based and arc-based hop-indexed formulations for the steiner tree problem with hop constraints. Technical report, Université libre de Bruxelles.

Fortz, B. and Poss, M. (2009). An improved benders decomposition applied to a multi-layer network design problem. *Operations Research Letters*, 37(5):359 – 364.

García, S. and Marín, A. (2020). Covering location problems. In Laporte, G., Stefan, N., and Saldanha da Gama, F., eds., *Location Science*, pages 99–119. Springer.

García-Archilla, B., Lozano, A. J., Mesa, J. A., and Perea, F. (2013). Grasp algorithms for the robust railway network design problem. *Journal of Heuristics*, 19(2):399–422.

Girgis, M. R., Mahmoud, T. M., Abdullatif, B. A., and Rabie, A. M. (2014). Solving the wireless mesh network design problem using genetic algorithm and simulated annealing optimization methods. *International Journal of Computer Applications*, 96(11).

González-Blanco, N., Lozano, A. J., Marianov, V., and Mesa, J. A. (2021). An integrated model for rapid and slow transit network design (short paper). In *21st Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Guignard, M. (2003). Lagrangean relaxation. *TOP*, 11(2):151–200.

Guihaire, V. and Hao, J.-K. (2008). Transit network design and scheduling: A global review. *Transportation Research Part A: Policy and Practice*, 42(10):1251–1273.

Hakimi, S. L. (1965). Optimum distribution of switching centers in a communication network and some related graph theoretic problems. *Operations Research*, 13(3):462–475.

Hakimi, S. L., Schmeichel, E. F., and Labbé, M. (1993). On locating path-or tree-shaped facilities on networks. *Networks*, 23(6):543–555.

Halpern, J. (1976). The location of a center-median convex combination on an undirected tree. *Journal of Regional Science*, 16(2):237–245.

Halpern, J. (1978). Finding minimal center-median convex combination (cent-dian) of a graph. *Management Science*, 24(5):535–544.

Hansen, P., Labbé, M., and Thisse, J.-F. (1991). From the median to the generalized center. *RAIRO-Operations Research-Recherche Opérationnelle*, 25(1):73–86.

Hellman, F. (2013). Sioux falls variants for network design. `http://www.bgu.ac.il/~bargera/tntp/SiouxFalls_CNDP/SiouxFallsVariantsForNetworkDesign.html`. Accessed November 24th, 2022.

Holland, J. H. (1975). Adaptation in natural and artificial systems. Univertity of Michigan Press, Ann Arbor, Mich.

Hutson, V. A. and ReVelle, C. S. (1989). Maximal direct covering tree problems. *Transportation Science*, 23(4):288–299.

Ji, W., Chen, Y., Chen, M., Chen, B.-W., Chen, Y., and Kung, S.-Y. (2015). Profit maximization through online advertising scheduling for a wireless video broadcast network. *IEEE Transactions on Mobile Computing*, 15(8):2064–2079.

Johnson, D. S. and Garey, M. R. (1979). *Computers and intractability: A guide to the theory of NP-completeness.* WH Freeman, U S A.

Johnson, D. S., Lenstra, J. K., and Rinnooy Kan, A. H. G. (1978). The complexity of the network design problem. *Networks*, 8(4):279–285.

Kermansshahi, S., Shafahi, M., Mollanejad, Y., and Zangui, M. (2010). Rapid transit network design using simulated annealing. *12th WCTR*, pages 1–15.

Kirkpatrick, S., Gelatt, C. D., and Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598):671–680.

Kleinert, T., Labbé, M., Ljubić, I., and Schmidt, M. (2021). A survey on mixed-integer programming techniques in bilevel optimization. *EURO Journal on Computational Optimization*, 9:100007.

Klincewicz, J. G. (1998). Hub location in backbone/tributary network design: a review. *Location Science*, 6(1-4):307–335.

Koster, A., Phan, T. K., and Tieves, M. (2013). Extended cutset inequalities for the network power consumption problem. *Electronic Notes in Discrete Mathematics*, 41:69–76.

Król, A. and Król, M. (2019). The design of a metro network using a genetic algorithm. *Applied Sciences*, 9(3):433.

Laporte, G. and Mesa, J. A. (2020). The design of rapid transit networks. In Laporte, G., Nickel, S., and Saldanha da Gama, F., eds., *Location Science*, chapter 24, pages 685–701. Springer.

Lee, K. Y. and El-Sharkawi, M. A. (2008). *Modern heuristic optimization techniques: theory and applications to power systems*, volume 39. John Wiley & Sons, New Jersey (U S A).

Ljubić, I., Mouaci, A., Perrot, N., and Gourdin, É. (2019). Benders decomposition for a node-capacitated virtual network functions placement and routing problem.

Ljubić, I., Putz, P., and Salazar-González, J.-J. (2012). Exact approaches to the single-source network loading problem. *Networks*, 59(1):89–106.

López-de-los Mozos, M. and Mesa, J. (1992). Location of cent-dian paths in tree graphs. In *Proceedings of the VI Meeting of the Euro Working Group on Locational Analysis*, pages 135–145.

Magnanti, T. L., Mireault, P., and Wong, R. T. (1986). Tailoring Benders decomposition for uncapacitated network design. In *Netflow at Pisa*, pages 112–154. Springer.

Magnanti, T. L. and Wong, R. T. (1981). Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Annals of Operations Research*, 29(3):464–484.

Magnanti, T. L. and Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science*, (18):1–55.

Mahéo, A. et al. (2020). Benders and its sub-problems.

Marín, Á. G. and Jaramillo, P. (2009). Urban rapid transit network design: accelerated Benders decomposition. *Annals of Operations Research*, 169(1):35–53.

Martin, R. K. (2012). *Large scale linear and integer optimization: a unified approach.* Springer Science & Business Media, New York (U S A).

Martins de Sá, E., Contreras, I., Cordeau, J.-F., Saraiva de Camargo, R., and de Miranda, G. (2015). The hub line location problem. *Transportation Science*, 49(3):500–518.

Mauttone, A., Cancela, H., and Urquhart, M. E. (2021). Public transportation. In Crainic, T. G., Gendreau, M., and Gendron, B., eds., *Network Design with Applications to Transportation and Logistics*, pages 539–565. Springer.

Maya Duque, P. and Sörensen, K. (2011). A grasp metaheuristic to improve accessibility after a disaster. *OR Spectrum*, 33(3):525–542.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems. *Mathematical Programming*, 10(1):147–175.

Mehta, D., O'Sullivan, B., Ozturk, C., and Quesada, L. (2015). An adaptive large neighbourhood search for designing transparent optical core network. In *2015 13th International Conference on Telecommunications (ConTEL)*, pages 1–8. IEEE.

Melkote, S. and Daskin, M. S. (2001). An integrated model of facility location and transportation network design. *Transportation Research Part A: Policy and Practice*, 35(6):515–538.

Mesa, J. A. (2018). Comments: on extensive facility location problems on networks: an updated review 1. *TOP*, 6(2):227–228.

Mesa, J. A. and Boffey, T. B. (1996). A review of extensive facility location in networks. *European Journal of Operational Research*, 95(3):592–603.

Mesa, J. A., Puerto, J., and Tamir, A. (2003). Improved algorithms for several network location problems with equality measures. *Discrete Applied Mathematics*, 130(3):437–448.

Monticelli, A. J., Romero, R., and Nobuhiro Asada, E. (2008). Fundamentals of simulated annealing. In Lee, K. Y. and El-Sharkawi, M. A., eds., *Modern Heuristic Optimization Techniques*, pages 123–146. John Wiley & Sons.

Norman, R. Z. and Rabin, M. O. (1959). An algorithm for a minimum cover of a graph. *Proceedings of the American Mathematical Society*, 10(2):315–319.

Ogryczak, W. (1997). On cent-dians of general networks. *Location Science*, 5(1):15–28.

O'Kelly, M. E. and Miller, H. J. (1994). The hub network design problem: a review and synthesis. *Journal of Transport Geography*, 2(1):31–40.

Pedersen, M. B., Crainic, T. G., and Madsen, O. B. (2009). Models and tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science*, 43(2):158–177.

Perea, F., Menezes, M. B., Mesa, J. A., and Rubio-Del-Rey, F. (2020). Transportation infrastructure network design in the presence of modal competition: computational complexity classification and a genetic algorithm. *TOP*, 28:442–474.

Potvin, J.-Y. and Gendreau, M. (2019). *Handbook of Metaheuristics*. Springer, New York (U S A).

Puerto, J., Ricca, F., and Scozzari, A. (2009). Extensive facility location problems on networks with equity measures. *Discrete Applied Mathematics*, 157(5):1069–1085.

Puerto, J., Ricca, F., and Scozzari, A. (2018). Extensive facility location problems on networks: an updated review. *TOP*, 26(2):187–226.

Rahmaniani, R., Crainic, T. G., Gendreau, M., and Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research*, 259(3):801–817.

Risso, C. and Robledo, F. (2013). Using grasp for designing a layered network: a real ip/mpls over dwdm application case. *International Journal of Metaheuristics*, 2(4):392–414.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472.

Schmidt, M. and Schöbel, A. (2014). Location of speed-up subnetworks. *Annals of Operations Research*, 223(1):379–401.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., and Dueck, G. (2000). Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2):139–171.

Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, 46.

Sinnl, M. and Ljubić, I. (2016). A node-based layered graph approach for the steiner tree problem with revenues, budget and hop-constraints. *Mathematical Programming Computation*, 8(4):461–490.

Smith, J. and Winter, P. (1991). *Topological Network Design, Proceedings of the NATO Advanced Research Workshop, Copenhagen 1989, Annals of Operations Research 33*. J.C. Baltzer AG.

Steuer, R. E. (1986). Multiple criteria optimization. *Theory, Computation and Applications.*

Toregas, C., Swain, R., ReVelle, C., and Bergman, L. (1971). The location of emergency service facilities. *Operations Research*, 19(6):1363–1373.

Wong, R. T. (2021). Telecommunications network design: Technology impacts and future directions. *Networks*, 77(2):205–224.

Wu, X., Lü, Z., and Glover, F. (2020). A matheuristic for a telecommunication network design problem with traffic grooming. *Omega*, 90:102003.

Xu, J., Chiu, S. Y., and Glover, F. (1996). Probabilistic tabu search for telecommunications network design. *Combinatorial Optimization: Theory and Practice*, 1(1):69–94.

Yaman, H. (2006). *Concentrator location in telecommunications networks*, volume 16. Springer Science & Business Media, New York (U S A).

Zhang, Y., Atasoy, B., and Negenborn, R. R. (2022). Preference-based multi-objective optimization for synchromodal transport using adaptive large neighborhood search. *Transportation Research Record*, 2676(3):71–87.