Programa de doctorado "Matemáticas"

PhD Dissertation

Contributions to robust and bilevel optimization models for decision-making

Author

Marina Leal Palazón

Supervisors

Prof. Dr. *Eduardo Conde Sánchez*

Prof. Dr. *Justo Puerto Albandoz*

February 22, 2019

A mis padres y mi hermana.

A Jose.

Agradecimientos

La vida es cuestión de decisiones. Y esta tesis es la consecuencia de una de las mías. Y de más gente.

Hacer un doctorado nunca estuvo entre mis planes. Creo que pocas veces lo había contemplado como una de las opciones. Se convirtió en opción tras un correo que Alfredo Marín me reenvío, en el que buscaban a un estudiante con este propósito. Gracias Alfredo por mostrarme esta puerta que tantas cosas buenas me ha traído cruzarla. Estos cuatro años de doctorado no sólo han sido horas tratando de comprender artículos, escribir modelos o implementar código, que también; han sido aprender de la mano de matemáticos, compartiendo horas delante de folios en blanco encuadernados o de un montón de folios en sucio. Han sido aprender viajando, escuchando el trabajo de otros y compartiendo el mío.

Gracias Justo y Eduardo por guiarme, con paciencia. Por sentaros conmigo las horas que hiciesen falta. Por vuestra disposición. Por discutir conmigo cosas que para vosotros ya hacía mucho que estaban claras. Gracias Eduardo por mis primeras conversaciones sobre investigación, por compartir conmigo la emoción de los primeros resultados que empiezan a salir, por tu confianza. Gracias Justo por mostrarme el mundo y al mundo de la investigación a través de tantos investigadores y en tantos lugares; para mí esta es una de las partes más bonitas de este trabajo; y gracias por abrirme esta puerta de la investigación a mí, y a tantos jóvenes.

Esta decisión me ha dado la oportunidad de vivir en sitios tan diferentes como Sevilla, Toronto o Bruselas, aprender de sus investigadores, disfrutar de su cultura y compartir con su gente. Gracias al proyecto y al departamento por acogerme con cariño. Al equipo de administración del IMUS por trabajar tan bien y con una sonrisa siempre. Thank you Igor and Oded for your generosity. Martine, merci pour ton hospitalité et ta disponibilité.

Estos cuatro años han sido también disfrutar. Por suerte, muchos otros decidieron, como yo, hacer su doctorado en Sevilla. Gracias por el día a día, por tantas risas, tantas comidas, tantas cenas, tantas fiestas, tanto hablar, tanto escuchar, tantos consejos, tanta paciencia, tanto apoyo. Estos cuatro años habéis sido mi otra familia, la familia del IMUS y la L2.

La parte menos bonita de cruzar esta puerta ha sido el tiempo que he dejado de dedicarles a los míos. A mi familia (la de casa, la Leal y la Palazón), a mis amigas de toda la vida, a mi Cris, a Esther, a Monta y a algunos más. Gracias por entenderlo. Por alegraros por la oportunidad que suponía para mí. Por, desde lejos, estar. Por hacerme partícipes, como podíais, de todo lo que os iba pasando. Gracias.

Gracias a mi abuela Rosa y a mi tía Mari por ser dos mujeres que me inspiran.

Jose, esta tesis también es tu decisión. Gracias. Gracias porque tu esfuerzo por esta tesis ha sido incluso mayor que el mío. La vida contigo es infinitamente (no numerable) mejor. Y gracias también a los tuyos por entenderlo.

Gracias a mis padres y mi hermana por creer en mí más que yo misma. Por estar siempre, para las grandes cosas y para las más pequeñas. Por vuestro apoyo. Gracias Lucía por decirme las cosas como son, por ayudarme mucho más de lo que crees. Gracias Cande y Tono por darme todo y más. Estar los cuatro siempre me hace feliz.

Resumen

Los problemas de optimización combinatorios han sido ampliamente estudiados en la literatura especializada desde mediados del siglo pasado. No obstante, en las últimas décadas ha habido un cambio de paradigma en el tratamiento de problemas cada vez más realistas, en los que se incluyen fuentes de aleatoriedad e incertidumbre en los datos, múltiples criterios de optimización y múltiples niveles de decisión. Esta tesis se desarrolla en este contexto. El objetivo principal de la misma es el de construir modelos de optimización que incorporen aspectos inciertos en los parámetros que definen el problema así como el desarrollo de modelos que incluyan múltiples niveles de decisión. Para dar respuesta a problemas con incertidumbre usaremos los modelos Minmax Regret de Optimización Robusta, mientras que las situaciones con múltiples decisiones secuenciales serán analizadas usando Optimización Binivel.

En los Capítulos 2, 3 y 4 se estudian diferentes problemas de decisión bajo incertidumbre a los que se dará una solución robusta que proteja al decisor minimizando el máximo regret en el que puede incurrir. El criterio minmax regret analiza el comportamiento del modelo bajo distintos escenarios posibles, comparando su eficiencia con la eficiencia óptima bajo cada escenario factible. El resultado es una solución con una eficiencia lo más próxima posible a la óptima en el conjunto de las posibles realizaciones de los parámetros desconocidos. En el Capítulo 2 se estudia un problema de diseño de redes en el que los costes, los pares proveedor-cliente y las demandas pueden ser inciertos, y además se utilizan poliedros para modelar la incertidumbre, permitiendo de este modo relaciones de dependencia entre los parámetros. En el Capítulo 3 se proponen, en el contexto de la secuenciación de tareas o la computación grid, versiones del problema del camino más corto y del problema del viajante de comercio en el que el coste de recorrer un arco depende de la posición que este ocupa en el camino, y además algunos de los parámetros que definen esta función de costes son inciertos. La combinación de la dependencia en los costes y la incertidumbre en los parámetros da lugar a dependencias entre los parámetros desconocidos, que obliga a modelar los posibles escenarios usando conjuntos más generales que los hipercubos, habitualmente utilizados en este contexto. En este capítulo, usaremos poliedros generales para este cometido. Para finalizar este primer bloque de aplicaciones, en el

Capítulo 4, se analiza un modelo de optimización en el que el conjunto de posibles escenarios puede ser alterado mediante la realización de inversiones en el sistema.

En los problemas estudiados en este primer bloque, cada decisión factible es evaluada en base a la reacción más desfavorable que pueda darse en el sistema. En los Capítulos 5 y 6 seguiremos usando esta idea pero ahora se supondrá que esa reacción a la decisión factible inicial está en manos de un adversario o follower. Estos dos capítulos se centran en el estudio de diferentes modelos binivel. La Optimización Binivel aborda problemas en los que existen dos niveles de decisión, con diferentes decisores en cada uno ellos y la decisión se toma de manera jerárquica. En concreto, en el Capítulo 5 se estudian distintos modelos de fijación de precios en el contexto de selección de carteras de valores, en los que el intermediario financiero, que se convierte en decisor, debe fijar los costes de invertir en determinados activos y el inversor debe seleccionar su cartera de acuerdo a distintos criterios. Finalmente, en el Capítulo 6 se estudia un problema de localización en el que hay distintos decisores, con intereses contrapuestos, que deben determinar secuencialmente la ubicación de distintas localizaciones. Este modelo de localización binivel se puede aplicar en contextos como la localización de servicios no deseados o peligrosos (plantas de reciclaje, centrales térmicas, etcétera) o en problemas de ataque-defensa.

Todos estos modelos se abordan mediante el uso de técnicas de Programación Matemática. De cada uno de ellos se analizan algunas de sus propiedades y se desarrollan formulaciones y algoritmos, que son examinados también desde el punto de vista computacional. Además, se justifica la validez de los modelos desde un enfoque de las aplicaciones prácticas. Los modelos presentados en esta tesis comparten la peculiaridad de requerir resolver distintos problemas de optimización encajados.

Abstract

Combinatorial optimization problems have been extensively studied in the specialized literature since the mid-twentieth century. However, in recent decades, there has been a paradigm shift to the treatment of ever more realistic problems, which include sources of randomness and uncertainty in the data, multiple optimization criteria and multiple levels of decision. This thesis concerns the development of such concepts. Our objective is to study optimization models that incorporate uncertainty elements in the parameters defining the model, as well as the development of optimization models integrating multiple decision levels. In order to consider problems under uncertainty, we use Minmax Regret models from Robust Optimization; whereas the multiplicity and hierarchy in the decision levels is addressed using Bilevel Optimization.

In Chapters 2, 3 and 4, we study different decision problems under uncertainty to which we give a robust solution that protects the decision-maker minimizing the maximum regret that may occur. This robust criterion analyzes the performance of the system under multiple possible scenarios, comparing its efficiency with the optimum one under each feasible scenario. We obtain, as a result, a solution whose efficiency is as close as possible to the optimal one in the set of feasible realizations of the uncertain parameters. In Chapter 2, we study a network design problem in which the costs, the pairs supplier-customer, and the demands can take uncertain values. Furthermore, the uncertainty in the parameters is modeled via polyhedral sets, thereby allowing relationships among the uncertain parameters. In Chapter 3, we propose time-dependent versions of the shortest path and traveling salesman problems in which the costs of traversing an arc depends on the relative position that the arc occupies in the path. Moreover, we assume that some of the parameters defining these costs can be uncertain. These models can be applied in the context of task sequencing or grid computing. The incorporation of time-dependencies together with uncertainties in the parameters gives rise to dependencies among the uncertain parameters, which require modeling the possible scenarios using more general sets than hypercubes, normally used in this context. In this chapter, we use general polyhedral sets with this purpose. To finalize this first block of applications, in

Chapter 4, we analyze an optimization model in which the set of possible scenarios can be modified by making some investments in the system.

In the problems studied in this first block, each feasible decision is evaluated based on the most unfavorable possible reaction of the system. In Chapters 5 and 6, we will still follow this idea, but assuming that the reaction to the initial feasible decision will be held by a follower or an adversary, instead of assuming the most unfavorable one. These two chapters are focused on the study of some bilevel models. Bilevel Optimization addresses optimization problems with multiple decision levels, different decision-makers in each level and a hierarchical decision order. In particular, in Chapter 5, we study some price setting problems in the context of portfolio selection. In these problems, the financial intermediary becomes a decision-maker and sets the transaction costs for investing in some securities, and the investor chooses her portfolio according to different criteria. Finally, in Chapter 6, we study a location problem with several decision-makers and opposite interests, that must set, sequentially, some location points. This bilevel location model can be applied in practical applications such as the location of semi-obnoxious facilities (power or electricity plants, waste dumps, etc.) or interdiction problems.

All these models are stated from a Mathematical Programming perspective, analyzing their properties and developing formulations and algorithms, that are tested from a computational point of view. Furthermore, we pay special attention to justifying the validity of the models from the practical applications point of view. The models presented in this thesis share the characteristic of involving the resolution of nested optimization problems.

Contents

Chapter 1

Introduction

Life is a matter of choices. Every day we face decision-making. Most of the decisions that we make routinely are insignificant but some others can be more relevant. When facing a decision, specially in these more significant ones, we undertake a careful analysis before deciding, with the aim of making the choice properly, optimally if it is possible. In this decision-making context, Operations Research (OR) brings us the theory and the tools to make better decisions. Among all theories in OR, Optimization is one of the most widely used ones. It drives us to optimal choices in challenging decision-making problems. Optimization is applied in many areas such as location, network design, shortest path, finance, game theory or pricing problems.

Optimization problems have been largely studied in the specialized literature; however, in the last decades, there has been a paradigm shift towards the treatment of more realistic problems, in which uncertainty sources in the data and multiple criteria or levels of optimization are included. This trend continues growing due to the economic importance of the existing applications in areas such as location, transportation, or planning. In this thesis, we study optimization models in which uncertainty affects the data, and optimization problems in which there exist several decisions makers, and thus multiple objectives, and a hierarchical decision order. We handle the uncertainty via the minmax regret criterion from Robust Optimization, and the hierarchical decision structure using Bilevel Optimization. In all our considered models, solving the optimization problem entails solving some other nested optimization problems.

1.1 Minmax regret and bilevel approaches for decision-making: relationship and theory

When trying to find an optimal decision, it is often possible to formulate the decision problem as one of maximization or minimization of one or several objective functions under a very wide collection of hypotheses. However, with the growth of theory and techniques and with the expansion of applications in Operations Research the variety in the structure of the problems in the field is immense. For instance, the objective function can be linear or nonlinear, nonlinear with differentiability or nonlinear without differentiability, the nature of the variables can be continuous or discrete, the objective of the problem and also the number of involved decision-makers can be single or multiple, some of the parameters of the problem can be uncertain or not, etcetera.

Among this large collection of configurations, we can find a considerable group of optimization problems, from very different nature, in which solving the optimization problem entails solving a nested optimization problem. That is, there exists a particular configuration in Optimization where one (or several) problem(s) is embedded

within another one. These problems can be understood as two (or more) stage problems: a first decision is made, in a first stage, taking into account that in a second (or latter) stage(s) other decision(s) will be made by *other*(s) decision-maker(s). Or equivalently, as problems in which the decision-maker integrates with her decision the decision process of another optimization problem. Examples of this type of problems can be found in areas such as Stochastic Optimization (Heyman and Sobel (2004); Shapiro et al. (2009); Tijms and Tijms (1994)), Robust Optimization (Ben-Tal et al. (2009); Kasperski (2008); Kouvelis and Yu (1997)) or Bilevel Optimization (Bard (2013); Dempe (2002a); Dempe et al. (2015)). We will focus on the last two.

Robust Optimization is focused on decision-making problems under uncertainty when no probabilistic information about the uncertain data is assumed. It is an effective way to structure this type of uncertainty and to make a decision in the presence of it. In Robust Optimization, the decision-maker constructs a solution that is feasible or *close* to the optimal for any realization of the uncertain parameters in a given set. In order to choose among the feasible solutions, there exist in the literature different criteria in order to assess the impact of the uncertainty on a given solution. Some of these criteria are based in the definition of what is called the worst-case performance of the assessed solution, such as the minmax, the minmax regret or the relative regret criteria. In these problems, the decision-maker chooses a solution anticipating the worst-case performance of the system for this solution. Sometimes, the worst-case performance of a solution can be related to actions of a real or figurative decision-maker who plays against our solution. That is, the decision-maker chooses an action taking into account that another real or figurative decision-maker, for instance, nature, will decide afterwards the actual performance of the system (the realized value of the uncertain parameters) in such a way that this is the worst possible realization of the uncertain parameters, for the chosen solution, under the selected criterion. The decision-maker acts in a first stage and the decision about the performance of the system is taken in a second stage. These problems are commonly formulated embedding the worst-case system performance decision problem within the decision-maker objective function.

On the other hand, Bilevel Optimization, as described in Bard (2013), addresses problems in which two decision-makers, each one with their own interests, act and react in a noncooperative, sequential manner. This hierarchical decision structure is written in terms of a mathematical program in which the second decision-maker's problem is part of the constraints of the first decision-maker's problem. Specifically, this second decision problem embedded within the constraints of the first one translates the fact that, some of the variables of the first problem are an optimal solution of the second decision problem. The first decision-maker anticipates the second decision-maker's choice. In these problems, there are clearly two decision-

makers, one acting in a first stage and another one in a second stage, and a nested optimization problem (within the constraints of the *main* problem).

In the following subsections, we provide a more detailed interpretation of the Robust and Bilevel Optimization, including a more in-depth discussion of formal definitions, applications, and results in the literature regarding these two approaches. In the robust case, we focus on a particular criterion, the minmax regret criterion, and in the bilevel case, we introduce also the widely used application to price setting problems.

1.1.1 Robust Optimization: the minmax regret criterion

Real-life is full of uncertainties. Hence, when attempting to build a realistic decision problem, the input data like costs, times, lengths, prices, etcetera, can be very commonly not exactly known in advance. This can be easily appreciated when thinking in concrete decision examples. For instance, when designing a new network, some costs such as those derived from the construction of new infrastructure can be uncertain; or when deciding the best schedule plans in which the cost of processing a task depends on its starting time, the execution times can be imprecise a priori; or in addition, when trying to determine a shortest path, the cost of traversing an arc of such path can be undetermined beforehand, or also the demands in supply chain problems can be unknown; and much more. Researchers became aware of the importance of accepting these uncertainties and managing them when modeling a problem long ago, as a consequence, literature is full of decision problems handling with uncertainty (see e.g. Abeledo et al. (2013); Averbakh and Berman (2003); Karasan et al. (2001); Conde (2004, 2007); Fortz et al. (2013); Marín et al. (2018); Merton (1969); Nickel et al. (2012); Olivares-Nadal and DeMiguel (2018); Pan and Nagi (2010); Puerto et al. (2009, 2011); Takahashi et al. (2001), etc.).

The uncertainty can be caused by a lack of information about the data or by the varying nature of the data itself. For instance, in the design of a network that has never been built in the past, the difficulty to estimate parameters (costs, times,...) can lie in the lack of historical data for similar projects, or even existing this data, the construction of new infrastructures can render the data obsolete; or for example, in a shortest path problem the travel times can vary depending on external factors as traffic. However, sometimes, the set defining the range of possible values for the uncertain parameters can be modeled in different forms depending on the available information. In fact, carrying out a proper study of the system, it may be possible to determine a more precise uncertainty set. For instance, in a supply chain problem in which the demand is uncertain, and a wide range of possible values is considered due to the expensiveness of obtaining tighter information, an investment to perform a prospective analysis of the demand may be sometimes carried out to obtain more

accurate values.

Whatever the nature and the structure of the uncertainty, the best way to handle it and to make decisions under its presence is to accept and assimilate this uncertainty and make an effort to incorporate it in the decision-making model (Kouvelis and Yu (1997)). Stochastic Programming and Robust Optimization are two of the most well-known and commonly used frameworks to manage uncertainty.

Stochastic Optimization is a natural and useful approach to address uncertainty when it is known the probability distribution of the input data. Under this knowledge, a typical approach is to select a decision that maximizes o minimizes an expected performance measure, under the assumed probability distribution. However, determining these probability distributions can be truly a difficult task, mainly when there exist multiple interdependent uncertain factors. Furthermore, an accepted drawback of Stochastic Programming is, as noticed by Kouvelis and Yu (1997) or Kasperski (2008), that the generated solutions can become meaningless when the process is not repeated several times since it is being optimized an expected performance of the system. Hence, the obtained solution may not be appropriate in problems where the solution is implemented just once; specially in those cases in which the decision-maker has to deal with the consequences on the system performance of the decision made under the realization of the parameters, as for instance when designing a network that will remain the same for a long period. Therefore, paying attention to the behaviour of the solution under any possible realization of the uncertain data can become essential.

In contrast, Robust Optimization can be a good approach for uncertainty environments where:

- There exists imprecise information about the probabilities of the possible realization of the parameters.
- The decision-maker is concerned about the effect of the solution with the realized data, and even existing some probabilities for the possible values, she desires to be prepared for any situation that may occur, not only for the most probable one but also for the least probable realization of the data. In these cases, the probabilistic information can be useful to select a subset of realizations sufficiently representative.
- The processes to implement are from a non-repetitive nature, that is, they are not going to be implemented periodically.
- The solutions are evaluated ex-post, that is, as if the actual realization of the parameters had been known before deciding.

The robust approach is based on the concept of scenario, that is, uncertainty is structured via a scenario-based description of the problem data. The notion of scenario refers to a concrete realization of the possible values of the unknown parameters. Then, as claimed before, the aim of a robust solution is to hedge against parameter variations, that is, to perform *well* under any of these plausible scenarios. In order to structure the uncertainty via scenarios two main representations have been considered in the literature: the discrete and the interval scenario cases. On the one hand, in the discrete approach, an explicit list of possible configurations of the parameters is given. This representation of the uncertainty allows considering correlations between the parameters; however, the number of possible scenarios must be finite or countable. On the other hand, in the interval case, it is assumed that each parameter can take on any value in a given interval, independently of the values taken by the other parameters. This interpretation of the uncertainty is a natural approach since in a lot of problems it is possible to determine a range of possible values for an uncertain parameter. Furthermore, this approach is not very data demanding. Nevertheless, the independence assumption among the uncertain data is sometimes not realistic or practicable. Thus, there exists a new tendency, still subtle, of allowing dependencies among the parameters in a continuous set of scenarios. This can be done by modeling uncertainty via more general sets of uncertainty. We will consider this uncertainty structure in some of the models we present.

Some of the most well-known robust approaches are the *absolute robust*, the *robust deviation* and the *relative deviation robust* criteria. The absolute robust criterion, generally known as minmax criterion, seeks a solution that minimizes the worst possible value among all the scenarios. A highly conservative approach quite useful in decision environments in which anticipating the reaction if the worst case happens is decisive. Less conservative and more compromised solutions are largely generated by the minmax regret criterion, also denoted as robust deviation. The regret of a solution under a scenario measures the cost or the risk of choosing such a solution under such scenario. It compares the performance of a solution with the optimal performance under a given scenario, that is, it compares each solution with the best decision that would have been taken if the scenario had been known in advance. Among all the regrets, one per scenario, the criterion takes the maximum one and tries to minimize it. Then, the minmax regret criterion minimizes the maximum deviation in the cost of a solution from the optimal one in each scenario. Finally, the relative robustness criterion compares, again, a solution with the optimal solution under a scenario, but measuring not the absolute deviation but the relative deviation respect to the optimal value under the scenario.

In Chapters 2, 3 and 4 we will adopt the minmax regret criterion to deal with the uncertainty. Since most of the minmax regret theory is developed for 0-1 determin-

istic combinatorial optimization problems, in the following subsection we formally define the minmax regret criterion and summarize some general results as well as complexity and approximation results for these type of problems. Most of this theory can be found, for instance, in Aissi et al. (2009) and the references therein.

Formal definition

Let us consider the class of 0-1 *deterministic combinatorial optimization problems* with linear objective function:

$$\begin{aligned} &\min \sum_{a \in A} c_a x_a \qquad (\mathcal{C}) \\ &\text{subject to} \\ &x \in X, \end{aligned}$$

where A is a finite set of elements and X a *feasible set* of solutions identified as subsets of A, that is, every $x \in X$ is a 0-1 (*characteristic*) vector with as many components as the number of elements in A, $x = (x_a)_{a \in A}$, where $x_a = 1$ if and only if a belongs to the subset represented by x; and $c_a \in \mathbb{Z}^+$ denotes a weight associated to every element $a \in A$ (cost, length, time, etc.).

Classical combinatorial optimization problems such as the discrete facility location, shortest path, knapsack, network design or minimum spanning tree problems fit in the above formulation $(\mathcal{C})$.

Let us assume now that coefficients c are uncertain. Let s denotes a given scenario and $c^s = (c^s_{a_1}, ..., c^s_{a_{|A|}})$ the values of the uncertain coefficients under such scenario. For the interval scenario case we consider that each coefficient c_a can take on any value in the interval $[c_a^-, c_a^+]$, independently of the rest, being $c_a^-, c_a^+ \in \mathbb{Z}^+$, $c_a^- \leq c_a^+$, $a \in A$. And let S be the set of all scenarios. Observe that in the interval scenario case, S is a hypercube determined by the Cartesian product of all these intervals.

Let $Z(x, s) = \sum_{a \in A} c^s_a x_a$ denote the value of a solution $x \in X$ under scenario $s \in S$. Let x^*_s denote an optimal solution under scenario $s \in S$, and $Z^*(s)$ the corresponding optimal value, that is

$$Z^*(s) = \min_{x \in X} Z(x, s) = Z(x^*_s, s).$$

In order to obtain the value $Z^*(s)$ the deterministic combinatorial optimization problem $\mathcal{C}$ must be solved for the fixed scenario $s \in S$.

Given a solution $x \in X$, its regret, $R(x, s)$, under scenario s, as underlined before, is given by the difference between the value of the solution minus the value of the

optimal solution under scenario s:

$$R(x,s) = Z(x,s) - Z^*(s).$$

The regret compares the behaviour of a solution with the best possible behaviour under a scenario s.

Since there exists one regret per each scenario, the maximum regret is an aggregation function that for a solution x takes the value of the maximum of the regrets among all scenarios:

$$R(x) = \max_{s\in S} R(x,s) = \max_{s\in S}\{Z(x,s) - Z^*(s)\}.$$

A scenario s that maximizes the right hand side is called a *worst case scenario* for x.

The minmax regret problem corresponding to $\mathcal{C}$ consists of finding a solution minimizing the maximum regret, which can be stated as

$$R^* = \min_{x\in X} R(x) = \min_{x\in X}\max_{s\in S}(Z(x,s) - Z^*(s)) = \min_{x\in X}\max_{s\in S} R(x,s). \qquad (\text{MINMAX REGRET } \mathcal{C})$$

This minmax regret criterion seeks a solution whose value is as close as possible to the optimal value for every scenario. The criterion can also be understood and stated as trying to set a threshold ε, as small as possible, looking for a solution x such that

$$Z(x,s) - Z^*(s) \leq \varepsilon, \quad \forall s \in S,$$

that is, looking for an ϵ solution under any possible scenario with ϵ as small as possible.

In order to present the theoretical results, we will denote by *DISCRETE* MINMAX REGRET $\mathcal{C}$ the MINMAX REGRET $\mathcal{C}$ problem in the discrete scenario case, and by *INTERVAL* MINMAX REGRET $\mathcal{C}$ in the interval scenario case.

General results

Solving a minmax regret problem implies solving, for each solution $x \in X$, a maximization problem over the set of scenarios. Sometimes an explicit form for this maximum can be obtained a priori by constructing a worst-case scenario. In particular, one of these scenarios can be set up for the *INTERVAL* MINMAX REGRET $\mathcal{C}$ Problem.

Proposition 1. *Given a minimization problem $\mathcal{C}$, the regret of a solution $x \in X$ is*

maximized for scenario w defined as follows:

$$c_a^w(x) = \begin{cases} c_a^+ & if\ x_a = 1, \\ c_a^- & if\ x_a = 0. \end{cases} \qquad a \in A.$$

Next result shows that an optimal solution of *INTERVAL* MINMAX REGRET $\mathcal{C}$ is an optimal solution of $\mathcal{C}$ for one of the extreme scenarios of S.

Proposition 2. *Given a minimization problem $\mathcal{C}$, an optimal solution x^* of* INTERVAL MINMAX REGRET $\mathcal{C}$ *corresponds to an optimal solution of $\mathcal{C}$ for at least one extreme of the hypercube of scenarios, in particular its most favourable scenario b_{x^*}, defined as follows:*

$$c_a^{b_{x^*}} = \begin{cases} c_a^- & if\ x_a^* = 1, \\ c_a^+ & if\ x_a^* = 0. \end{cases} \qquad a \in A.$$

Complexity results

We compile in the following Table 1.1, taken from Aissi et al. (2009), the complexity of some of the minmax regret versions of classical combinatorial problems. In the first column of Table 1.1 it is shown the name of the problem, in the second one, the complexity of such problems when the number of possible scenarios is constant, in the third column, the complexity for a non-constant number of scenarios, and in the last one, the complexity for the interval case.

PROBLEMS	CONSTANT NUMBER OF SCENARIOS	NON-CONSTANT NUMBER OF SCENARIOS	INTERVAL SCENARIO CASE
SHORTEST PATH	**NP-hard, pseudo-poly** (Yu and Yang (1998))	**Strongly NP-hard** (Kouvelis and Yu (1997))	**Strongly NP-hard** (Averbakh and Lebedev (2004))
SPANNING TREE	**NP-hard, pseudo-poly** (Kouvelis and Yu (1997); Aissi et al. (2006))	**Strongly NP-hard** (Aissi et al. (2007))	**Strongly NP-hard** (Averbakh and Lebedev (2004))
ASSIGNMENT	**NP-hard** (Kouvelis and Yu (1997))	**Strongly NP-hard** (Aissi et al. (2005a))	**Strongly NP-hard** (Aissi et al. (2005a))
KNAPSACK	**NP-hard, pseudo-poly** (Kouvelis and Yu (1997))	**Strongly NP-hard** (Aissi et al. (2007))	**NP-hard**
MIN CUT	**Polynomial** (Aissi et al. (2005b))	**Strongly NP-hard** (Aissi et al. (2005b))	**Polynomial** (Aissi et al. (2005b))
MIN s-t CUT	**Strongly NP-hard** (Aissi et al. (2005b))	**Strongly NP-hard** (Aissi et al. (2005b))	**Strongly NP-hard** (Aissi et al. (2005b))

Table 1.1: Complexity of minmax regret versions of classical combinatorial problems. (See Aissi et al. (2009)).

Table 1.1 shows the hardness of the minmax regret problem. It is interesting to observe that most of the graph problems that are polynomial-time solvable become

NP-hard in the minmax regret version; although the first of these polynomial-time solvable problem whose minmax regret version becomes strongly NP-hard (even for two scenarios) is the MIN s-t CUT.

Approximation results

Due to the difficulty of solving minmax regret problems it has been proposed in the literature some constant factor approximation results. Approximate solutions can be a valid approach to the solutions when obtaining an optimal one is not possible, or also a good starting point to find an optimal solution.

The idea to find these approximate solutions is to solve the deterministic problem $\mathcal{C}$ for a *nice* scenario. We gather here some approximation results.

The next proposition, given by Kouvelis and Yu (1997), provides an approximation result for the *DISCRETE* MINMAX REGRET $\mathcal{C}$ by solving the discrete problem for the mean scenario.

Proposition 3. *Consider an instance I of DISCRETE MINMAX REGRET $\mathcal{C}$ with k scenarios where each scenario $s \in S$ is represented by $(c^s_{a_1}, ..., c^s_{a_{|A|}})$. Consider also an instance I' of $\mathcal{C}$ where each coefficient of the objective function is defined by $c'_i = \sum_{s=1}^{k} \frac{c^s_a}{k}$, $a \in A$. Then x' an optimal solution of I' is such that $\max_{s\in S} R(x', s) \leq k \cdot opt(I)$, where $opt(I)$ denotes the optimal value of Problem MINMAX REGRET $\mathcal{C}$ under instance I.*

In fact, this approximate solution may be better than a k-approximation in some cases. In order to compute the maximum regret we aim to maximize the following function of c $\varphi(c) = \{\sum_{i=1}^{n} c_i x - \min_{y\in X} \sum_{i=1}^{n} c_i y\}$.

Since $\varphi(c)$ is a convex function, it is enough to consider S' the convex hull of S. Therefore, we only need to consider the v ($v \leq k$) scenarios that are extreme points of the convex hull of S S. In this way, if $v < k$ we obtain a v-approximation.

For the cases in which S is not a discrete set and it is a polyhedron, function $\varphi(c)$ will reach its optimal value in some vertex of the polyhedron, so it would be enough to consider the vertex of it. However, for the *INTERVAL* MINMAX REGRET $\mathcal{C}$ problem, if there are n uncertain parameters, the number of vertex of the polyhedron, which is a hypercube, is 2^n. Kasperski and Zielinski (2006) showed that we can obtain a better approximation for the *INTERVAL* MINMAX REGRET $\mathcal{C}$, namely, a 2-approximation.

Proposition 4. *Given an instance I of INTERVAL MINMAX REGRET $\mathcal{C}$, consider an instance I' of $\mathcal{C}$ where each coefficient of the objective function is defined by $c'_a = \frac{1}{2}(c^-_a + c^+_a)$ (I' is the mid-point scenario). Then x' an optimal solution of I' has $\max_{s\in S} R(x', s) \leq 2opt(I)$, where $opt(I)$ denotes the optimal value of instance I.*

This approximation result has been recently generalized under more generic assumptions. See for instance Chassein and Goerigk (2015); Conde (2012).

Existing methods

In this section we briefly review some exact procedures to solve minmax regret problems.

For the discrete scenario case Kouvelis and Yu (1997) implemented a branch and bound procedure to solve the minmax regret version of several combinatorial optimization problems. In this algorithm, a lower bound on each node of the tree is calculated using surrogate relaxation (Aissi et al. (2009)).

For the interval scenario case, it is common, when possible, to reformulate the non-linear minmax regret problem with a MILP formulation. It is possible to develop MILP formulations for zero duality gap $\mathcal{C}$ problems. These formulations are obtained by replacing the embedded $\mathcal{C}$ problem by its dual one and the maximum regret embedded problem by a worst-case scenario expression. This type of resolution was initially studied by Karasan et al. (2001); Yaman et al. (2001). We also apply it in Chapters 2, 3 and 4.

One of the most used procedures to deal with minmax regret version of problem $\mathcal{C}$ is a relaxation procedure based on a Benders decomposition; see for instance Montemanni and Gambardella (2004, 2005b); Aissi et al. (2009) and the references therein. Branch and bound algorithms were also designed for solving minmax regret version of some combinatorial problems; see e.g. Aron and Van Hentenryck (2002); Montemanni et al. (2004); Montemanni and Gambardella (2005a). Following these ideas we implement in Chapters 2, 3 and 4 Benders decomposition algorithms and branch and bound algorithms based on Benders cuts.

Preprocessing techniques has also been incorporated into the approaches described above. For further details we refer the reader to Karasan et al. (2001); Aissi et al. (2009); Catanzaro et al. (2011) and the references therein.

1.1.2 Bilevel Optimization

Multiobjective Optimization addresses problems with several simultaneous objectives; meanwhile, Game Theory deals with problems in which different decision-makers strategically interact. As noted in Bard (2013), Bilevel Optimization can be understood as the combination of these two. It targets optimization problem with two decision-makers, and hence two objective functions, and a hierarchical decision structure. One of the decision-makers, called the leader, plays first, and then, the other one, called the follower, observes the decision made by the leader and acts optimizing her choice. Bilevel Programming studies the problem from the leader's point of view, that is, studies the leader's problem. In this problem it is optimized the leader's strategy, minimizing (maximizing) the leader's objective function over the set of feasible strategies, taking into account that the follower always reacts op-

timally to the leader's decision. In these problems perfect information is assumed, each decision-maker knows the objective and the feasible strategies of the other.

The bilevel concept has been extended to multiple levels. Multilevel Programming addresses problems in which several (two or more) decision-makers interact hierarchically, and in each sublevel, the decision-maker observes upper levels decisions and then act, according to her own interests. A remarkable characteristic of multilevel problems, and thus of bilevel problems, is that each decision-maker is affected by the actions of the other decision-makers. A decision-maker from a particular level can modify with her choices the actions of others, but does not control them completely.

In many real situations decisions do not depend on a single viewpoint, and; in many of these occasions, the decision process can be modeled as a bilevel or multilevel problem. In fact, one of the greatest interest in Bilevel Optimization lies, apart from the challenging difficulty of these problems, in the large amount of actual applications. Consider for instance a problem in which a good must be placed in a particular location and must be protected from some attackers; the location decision can be made keeping in mind that, once it is located, the attackers will try to reach it. A lot of more applications can be found in the fields of location, credit allocation, production planning, electric power pricing, network design problems, security, economic game theory, etcetera. See for instance Arroyo (2010); Dempe et al. (2015); Camacho-Vallejo et al. (2014); Calvete et al. (2011); Colson et al. (2005); Griffith et al. (1998); Labbé and Violin (2013); Ma (2016); Marcotte (1986); Vicente and Calamai (1994) and the references therein.

Bilevel Programming is also a good framework to model Stackelberg games from game theory. These games were introduced by von Stackelberg et al. (1952). In the game there are two players, the leader and the follower, the leader plays first and decides her strategy taking into account the follower optimal reaction, and the follower plays second, knowing the leader's choice. A closed concept in game theory is the notion of Nash equilibrium. A solution in which any player has something to gain by changing only their own strategy. As noted in Labbé and Violin (2013), it is interesting to remark that the solutions or equilibrium obtained by these two approaches correspond to different assumptions in the game: Stackelberg solutions are obtained in hierarchical decision structure games, meanwhile Nash equilibriums come from simultaneous games, in which all players come at the same time.

Another powerful use of Bilevel Programming is in the so-called pricing problems. This field encompasses problems in which a first decision-maker, for instance, a company or a government, must fix the prices or taxes of some products so that its revenue is maximized, but the reaction of the potential clients or users must be taken into account: if the prices are too high the sales will be shorter. This relationship,

between the price-fixer and the consumers, can be modeled via Bilevel Optimization since there exists a hierarchical decision structure, and the values of the decision variables of one level influence the optimal solution of the other level (Labbé and Violin (2013)). Think for instance in the problem of a bank that must decide the transaction costs for clients investments; the bank can fix them trying to maximize its reward, but must take into account that these rates will condition clients' investments and by extension the bank benefit. Several examples of price setting problems modeled via Bilevel Optimization can be found for instance in the fields of toll optimization of highways, pricing of express mail delivery or passengers transportation systems, pricing telecommunications packages, etcetera (see for example Labbé and Violin (2013) and the references therein).

Formal definition

Let $x \in \mathbb{R}^{n_1}$ and $y \in \mathbb{R}^{n_2}$ denote the decision vectors, $F : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}$ and $f : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ the objective functions and $G : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{m_1}$ and $g : \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \to \mathbb{R}^{m_2}$ the vector-valued functions of the constraints of the leader and the follower problems, respectively. Let X represent the set of constraints concerning only the x variables. The general Bilevel Programming problem (BP) can be defined as:

$$\min_{x \in X, y} \; F(x, y) \tag{1.1}$$

$$\text{s.t. } G(x, y) \leq 0, \tag{1.2}$$

$$y \in arg\min_{y} f(x, y), \tag{1.3}$$

$$\text{s.t. } g(x, y) \leq 0. \tag{1.4}$$

(1.1) and (1.2) constitute the upper level problem, while (1.3) and (1.4) define the lower level problem. It can be observed that the lower level or follower problem is included in BP as a part of the constraints. That is, in BP constraints (1.3) and (1.4) translate the fact that some of the variables of the problem, the y variables, are an optimal solution of a second and nested optimization problem. The lower level problem must be solvable for global minima. In practice it is assumed to be convex (Colson et al. (2005)).

The bilevel and multilevel terms were firstly introduced by Candler and Norton (1977), however, as pointed out by Colson et al. (2005), bilevel problems were introduced before under the designation of *mathematical programs with optimization problems in the constraints* in a series of papers by Bracken and McGill (1973, 1974, 1978), developed in the context of military and production and marketing decision-making. In game theory, the Bilevel Programming problems were introduced by the

name of Stackelberg games.

The above BP formulation can be particularly adapted for price setting problems. Let us assume that there are t_1 taxed and t_2 untaxed activities, $T \in \mathbb{R}^{t_1}$ is the tax vector, $x \in \mathbb{R}^{t_1}$ and $y \in \mathbb{R}^{t_2}$ are the decision vectors associated with taxed and untaxed activities, respectively, and F and f are the objective functions of the leader and the follower objective values. And let $g(x, y)$ be the vector-valued function of the constraints. The price setting bilevel problem (**P-BP**) can be formulated as the following bilevel problem:

$$\begin{aligned} \max_{T} \; & F(T, x, y) \\ & (x, y) \in arg \min_{x,y} f(T, x, y), \\ & \text{s.t. } g(x, y) \leq 0. \end{aligned}$$

In (**P-BP**) a function of the revenue of the price-fixer is maximized assuming that, the taxed and untaxed activities are chosen by the user afterwards trying to minimize a function of the cost according to some constraints.

We define now different concepts related to Bilevel Programming.

The *relaxed problem* associated to BP consists of removing the restriction that the y variables must be an optimal solution of the follower level problem, that is, removing constraint (1.3). Then, the *relaxed feasible region* or *constrained region* is defined as

$$\Omega = \{(x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} : x \in X, G(x, y) \leq 0 \text{ and } g(x, y) \leq 0\}.$$

For a given vector $\bar{x} \in X$, the *lower level feasible set* is

$$\Omega(\bar{x}) = \{y \in \mathbb{R}^{n_2} : g(\bar{x}, y) \leq 0\},$$

meanwhile the *lower level reaction set* is

$$LR(\bar{x}) = \{y \in \mathbb{R}^{n_2} : y \in arg \min\{f(\bar{x}, \hat{y}) : \hat{y} \in \Omega(\bar{x})\}\}.$$

Every $y \in LR(\bar{x})$ is a *rational response*.

Finally, the *induced region*, which include all the feasible points of the BP and corresponds to the feasible set of the leader, is defined as

$$\mathcal{IR} = \{(x, y) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} : x \in X, G(x, y) \leq 0 \text{ and } y \in LR(x)\}.$$

This is usually nonconvex and it can be disconnected or even empty.

In a BP problem, for a given value of x variables, the follower problem may have multiple optimal solutions. Therefore, if it happens, different choosing attitudes can be adopted by the follower. The follower's behavior will give rise to different modelling approaches. A cooperative attitude leads to an optimistic solution so that when there exist multiple follower solutions, the leader assumes that the follower will choose the most favourable one for the leader. Formally, when $LR(x)$ is not a singleton, a point $(x^*, y^*) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ is called a *local optimistic solution* for problem BP if

$$\begin{aligned} & x^* \in X, \\ & G(x^*, y^*) \leq 0, \\ & y^* \in LR(x^*), \\ & F(x^*, y^*) \leq F(x^*, y), \quad \forall y \in LR(x^*), \end{aligned}$$

and there exists an open neighborhood of x^*, with radius $\delta > 0$, $B(x^*, \delta)$, such that

$$\phi_0(x^*) \leq \phi_0(x), \ \forall x \in B(x^*, \delta) \cap X,$$

where $\phi_0(x) = \min_y \{F(x, y) : y \in LR(x)\}$. It is called a *global optimistic solution* if $\delta = \infty$ can be chosen.

In contrast, an aggressive attitude leads to a pessimistic solution in which the leader assumes that the follower will choose, among the optimal solutions for the follower, the worst possible one for the leader. A point $(x^*, y^*) \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ is called a *local pessimistic solution* for problem BP if

$$\begin{aligned} & x^* \in X, \\ & G(x^*, y^*) \leq 0, \\ & y^* \in LR(x^*), \\ & F(x^*, y^*) \geq F(x^*, y), \quad \forall y \in LR(x^*), \end{aligned}$$

and there exists an open neighborhood of x^*, with radius $\delta > 0$, $B(x^*, \delta)$, such that

$$\phi_p(x^*) \leq \phi_p(x), \ \forall x \in B(x^*, \delta) \cap X,$$

where $\phi_p(x) = \max_y \{F(x, y) : y \in LR(x)\}$. It is called a *global pessimistic solution* if $\delta = \infty$ can be chosen. For further details we refer the reader to Dempe (2002b); Loridan and Morgan (1996)

In the bilevel problems that we present, Chapters 5 and 6, we will only assume cooperative behaviour.

As referred before, the Bilevel Programming concept can also be extended to Multilevel by considering more than two levels, that is, more than two decision-makers playing hierarchically and hence, more than one nested problem. Another realistic and useful expansion is to consider that there exist several decision-makers in the same follower level. All these followers observe the leader's decision and react, all at the same time, according to their own interests. The decision of a follower can depend, or not, on the other followers decisions. This extension will be considered in Chapters 5 and 6.

General results

We gather in this subsection some basic properties and results for linear and binary bilevel problems. These results and the proofs or references can be found in Bard (2013), which can be also used for further details and references.

Linear Bilevel Programming includes all bilevel problems in which all the functions, constraints and variables are linear. The following results hold for this type of problems.

Theorem 1. *The induced region $\mathcal{IR}$ of a linear bilevel problem can be written equivalently as a piecewise linear equality constraint comprised of supporting hyperplanes of Ω.*

From this theorem, the next two results follow.

Corollary 1. *The linear BP is equivalent to minimizing F over a feasible region comprised of a piecewise linear equality constraint.*

Corollary 2. *A solution to the linear BP occurs at a vertex of $\mathcal{IR}$.*

The following results show that a solution vertex of $\mathcal{IR}$ is also a vertex of Ω.

Theorem 2. *The solution (x^*, y^*) of the linear BP occurs at a vertex of Ω.*

Corollary 3. *If x is an extreme point of $\mathcal{IR}$, then it is an extreme point of Ω.*

We consider now the zero-one linear BP, where all or some variables are restricted to binary variables.

Let us denote by (01-BP) the case in which all variables are binary and by (01U-BP) and (01L-BP) the cases in which the upper level and lower level variables, respectively, are binary.

The following results address the existence of optimal solutions

Theorem 3. *For the linear BP, let Ω be a bounded set, i.e., a polytope. If $\Omega_u := \{(x,y) : G(x,y) \leq 0)\} = \mathbb{R}^{n_1+n_2}$, then linear BP, (01-BP) and (01U-BP) have an optimal solution if $\Omega \neq \emptyset$. If $\Omega_u \neq \mathbb{R}^{n_1+n_2}$, then linear BP, (01-BP) and (01U-BP) have an optimal solution if there exists an $\bar{x} \in X$ such that $(\bar{x}, \bar{y}) \in \Omega_u$.*

This theorem provided sufficient conditions for the existence of optimal solutions when Ω is bounded. These conditions are equivalent to the conditions under which the induced region is nonempty.

The latter result, provide conditions to guarantee that an existing optimal solution of (01L-BP) is a boundary point of the set Ω.

Theorem 4. *Let $\Omega_u := \{(x,y) : G(x,y) \leq 0)\} = \mathbb{R}^{n_1+n_2}$, and $\Omega \neq \emptyset$ and suppose that there exists an optimal solution (x^*, y^*) to (01L-BP). Then (x^*, y^*) is a boundary point of Ω.*

Complexity results

Bilevel problems are, from a computational point of view, very difficult to solve. The linear BP was first shown to be NP-hard by Jeroslow (1985). Hansen et al. (1992) strengthened this result by proving that the linear minmax bilevel problem is strongly NP-hard. An immediate consequence of it was the strongly NP-hardness of the linear BP problem. Later, Vicente et al. (1994) proved that even the problems of checking strict local optimality and local optimality in linear BP are NP-hard.

Existing methods

Due to the difficulty of the bilevel problems it is not surprising that most of the techniques and algorithms on the topic have been focused on the *easiest* bilevel programs, those with nice properties such as linear, quadratic or convex objective function and/or constraints functions. Even so, over the years, more complex bilevel problems, for instance, with discrete variables or nonlinear programming problems in both levels have also been subject of research (Colson et al. (2005)).

We briefly gather in this section some of the more successful schemes to deal with bilevel problems. For further details and references on the highlighted methods and other ones we refer the reader to Bard (2013); Colson et al. (2005) and the references therein.

For the case of linear BP, making use of Theorem 2, whereby there exist an optimal solution of the problem in the vertex of Ω, a wide class of methods for solving this linear BP have been developed based on vertex enumeration in the context of the simplex method. For instance, the *Kth-Best Algorithm*, proposed by Bialas and Karwan (1982), systematically explores the vertex beginning with

the basis associated with the optimal solution to the linear program obtained by removing constraint (1.3). The methods stops with a global optima.

Other method, possibly the most popular one, applicable when the lower level problem is convex and regular, is the *Khun-Tucker approach*. We will apply it in Chapters 4 and 5. The first step is to replace the lower level problem by its Karush-Kuhn-Tucker (KKT) conditions, giving rise to a single level reformulation of the BP. This method is also known as *Branch-and-bound*, since the basic idea is to apply a branch and bound strategy to deal with the complementary constraint.

Alternative approaches proposed in the literature for linear BP are methods based on penalty functions, complementary pivoting or reverse convex programming, or also trying to find an explicit expression for the optimal solutions of the lower-level problem.

Methods for Mixed Integer Bilevel Linear Problems (MIBLP) have also been studied, in fact, it is an appearing current subject of research. The most common approaches to solve this type of problems, under some conditions, are Branch and Bound (see e.g. Moore and Bard (1990)) and Branch and Cut algorithms (see e.g. DeNegre and Ralphs (2009); Fischetti et al. (2018)). Very recently, Fischetti et al. (2017) proposed a new general-purpose branch and cut exact algorithm for MIBLP, with continuous and discrete variables in both levels, based on several new classes of valid inequalities. With this new algorithm, they were able to solve to optimality more than 300 previously unsolved instances from the literature. In addition, decomposition techniques, such as Benders Decomposition, can also be used for MIBLP under some particular conditions. For instance, we will apply an algorithm based on a Benders decomposition to a MIBLP in Chapter 6.

Due to the inherent complexity of the Bilevel Optimization, several heuristics have also been developed in the literature to handle BP (see e.g. Bard (2013) and the references therein).

The minmax regret problem as a bilevel model

As said before, in the minmax regret models, a solution is chosen taking into account the most unfavorable possible reaction of the system. That is, the system *chooses*, for each feasible solution, the worst possible realization of the uncertain parameters. This can be understood, from the Bilevel Optimization point of view, as the reaction of the system is determined by a follower, who acts in this case against the leader. Hence, the minmax regret model

$$R^* = \min_{x \in X} \max_{s \in S} R(x, s) = \min_{x \in X} \max_{\substack{s \in S \\ y \in X}} Z(x, s) - Z(y, s),$$

can be rewritten as the following bilevel programming problem

$$
\begin{aligned}
R^* = \min\ & R(x,y,s) \\
& \text{s.t. } x \in X, \\
& \quad (y,s) \in arg\max R(x,y,s), \\
& \quad\quad \text{s.t. } s \in S, \\
& \quad\quad\quad y \in X,
\end{aligned}
$$

where $R(x,y,s) = Z(x,s) - Z(y,s)$.

1.2 Contributions of this thesis

The goal of this thesis is to derive optimization models, in the context of decision-making, in which uncertainty is incorporated in the parameters defining the problem; and also to build models in which there exist two (or more) decision-makers, with two (or more) objective functions, and a hierarchical decision structure. To give a compromise solution in those problems with uncertain parameters the minmax regret paradigm is considered, whereas the hierarchy in the decision process is managed via Bilevel Optimization. Hence, two main blocks can be distinguished in the thesis, one devoted to the contributions to minmax regret models, addressed in Chapters 2, 3 and 4, and another one with the contributions to bilevel models, addressed in Chapters 5 and 6. In the problems studied in the first chapters, each feasible decision is evaluated with respect to the most unfavorable possible reaction in the system. In Chapters 5 and 6 we continue using this idea but assuming in this case that, the reaction to the initial feasible decision is carried out by an adversary of a follower. All the models we propose, the minmax regret and the bilevel ones, belong to a class of optimization problems in which solving the optimization problem entails solving some other nested optimization problems. In all these models, formulations, algorithms, and properties of the problems are stated from a Mathematical Programming perspective. The validity of all models is justified from the practical applications point of view and we also analyze computationally the viability of the proposed formulations and optimization algorithms.

In Chapter 2, based on the paper by E.Conde and M.Leal (Conde and Leal, 2019), we address the problem of designing a robust supply network, assuming uncertain some of the parameters defining the problem. We examine the cases in which the uncertainty affects the construction and/or operating costs, and/or the existing set of supplier-client pairs, and/or the demand to serve. Our contributions are modeling

the uncertainty using the minmax regret criterion, and assuming the uncertainty set of parameters is given as a polyhedron, allowing in this way dependence relationships between the uncertain parameters. Benders decomposition techniques are used to solve the models and the proposed algorithms are tested in a computational experiment.

In Chapter 3, based on the paper by E.Conde, M.Leal and J.Puerto (Conde et al., 2018), we study the minmax regret version of a time-dependent shortest path problem. In particular, we assume that the cost structure takes into account the position that each arc occupies in the path. Furthermore, we assume uncertainty in the costs defining the time-dependency. Our first contribution in this chapter is combining the time-dependency and the uncertainty of the costs in the context of the minmax regret criterion. As a consequence of the combination, relationships between the uncertain parameters emerge, and the common independence assumption among the set of costs does not hold. In order to deal with the model, we provide a MILP formulation for the uncertainty interval case, extending the classical duality and worst-case scenario developments to obtain MILP formulations for minmax regret problems, which represents another contribution of this chapter. Then, we extend the model to a new and more general case in which polyhedral sets of uncertainty are considered. This approach can also be extended to the minmax regret time-dependent Traveling Salesman Problem. We propose three different algorithms based on Benders decomposition to solve the problem. We reinforce the algorithms with constant factor approximations. The formulation and algorithms are tested in a computational experiment.

In Chapter 4, based on the paper by E.Conde and M.Leal (Conde and Leal, 2017), we propose a new minmax regret optimization model in which the decision-maker has some control over the uncertain parameters of the system. Until now in the literature, most of the minmax regret models have ignored the possibility of changes in the uncertainty, or in the degree of knowledge about the attainable values of the parameters due to actions carried out in the system. Hence, in our model, as a contribution, we incorporate this possibility by assuming that in some cases, spending some resources we may be able to change the cost structure of the system and take advantage of it to find a robust solution. Some properties of the model allow us to construct Mathematical Programming formulations that can be solved by off-the-shelf solvers. We illustrate the model in the shortest path problem in a network in which investments in the system produce variations in the uncertainty intervals for the arc costs. We extend some existing results on constant factor approximation for minmax regret problems without investments, give an approximate algorithm and conduct a computational study.

In Chapter 5, based on the paper by M.Leal, D.Ponce and J.Puerto (Leal et al.,

2018), we propose various price setting models in the context of Portfolio Optimization. We present different bilevel portfolio selection models in which the transaction costs, generally assumed given in the portfolio problems, become decision variables, and hence, the financial intermediary becomes a decision-maker in the problem. This financial intermediary decides on the unit transaction costs for investing in some securities, maximizing its benefits, and the investors choose the portfolio, minimizing risk and ensuring a desired expected return. Therefore, our contributions are turning transaction costs into decision variables, and incorporating two levels of decision-makers in the portfolio problem. We present different bilevel programming versions, depending on who makes the decision first, the investor or the financial institution, and also analyze the social welfare version. We study the different models, analyze some properties, develop MILP formulations and algorithms and report some computational experiments performed on real data taken from the IBEX 35.

In Chapter 6, based on the paper by M.Labbé, M.Leal and J.Puerto (Labbé et al., 2018), we address a new bilevel location model motivated by real-life problems in which the location decision generates disagreements among users with opposite interests. We consider two different levels of decision-makers. The first one chooses among a number of fixed potential locations some primary facilities to set up; next, the second one chooses the location of a secondary facility in a continuous framework. The leader and the follower have opposite targets; the leader's and follower's goal is to maximize and minimize, respectively, some proxy of the overall weighted distance between the primary and secondary facilities. We develop the bilevel model for one follower and any polyhedral distance, proposing a proof of the NP-hardness of the problem, different MILP formulations and algorithms; and later we extend it for several followers and any ℓ_p-norm, $p \in \mathbb{Q}$, $p \geq 1$. We also report some computational results.

Finally, in Chapter 7, conclusions and some possible future research lines for the different proposed models are briefly discussed.

Chapter 2

A minmax regret model for the design of supply networks under uncertainty

In this chapter we consider an optimization model to find a robust design of supply networks under uncertainty. We explore the cases in which the uncertainties affect, simultaneously or not, the construction and/or operating costs, the pairs supplier-receptor, and the demands. The data uncertainty is modelled through polyhedral sets, allowing in this way describing dependency relations between the uncertain parameters, and this uncertainty is incorporated and handled in the model using the minmax regret criterion. In order to solve the model we use Benders decomposition methodology. Finally, we perform a computational experiment using specialized optimization solvers to test the efficiency of the numerical algorithms.

2.1 Introduction

Finding efficient designs of a distribution network through which the demands of certain commodities must be served has been an interesting optimization problem in the last decades. Once such an infrastructure is built, it will be, in general, very difficult and costly to modify its design. Therefore, it is important to take into account not only the current construction costs, but also all the elements that may affect the efficiency of the system during a reasonable period of its service life. Some of these elements could be costs of routing goods (Gutiérrez et al. (1996); Peng et al. (2011)), link maintenance (roads, pipelines, electrical connections, optical fibre, etc) or demands supplied by the system (Jabbarzadeh et al. (2017); Lee et al. (2013); Pan and Nagi (2010)).

However, in many cases, at the moment in which the network is designed, some of this data could remain unknown (Gutiérrez et al. (1996); Jabbarzadeh et al. (2017); Lee et al. (2013); Magnanti and Wong (1984); Pan and Nagi (2010)) and, furthermore, it could be unrealistic to estimate data. As referred in Chapter 1, in certain cases, the difficulty lies in the lack of historical data to estimate costs or demands under which the future behaviour of the system will be evaluated. Even though this data exists, it could become obsolete for effective estimations due to the very same fact of the construction of the new infrastructure. Indeed, the stability conditions of the data sample, frequently required in the estimation procedures, as the identical distribution, could be seriously compromised in the new framework. In other situations, the system could be affected by rare events, like facility disruptions (Fereiduni and Shahanaghi (2017); Peng et al. (2011)), which modify the input data. In these cases, it could be nonsense to try to estimate the consequences of the event in the data due to its unpredictable nature. However, it seems to be reasonable to protect somehow the network efficiency from the negative effects of these events that could have considerable economical consequences in supply chain infrastructures (Hendricks and Singhal (2005)).

In order to protect the performance of the system, we can use an interesting property observed by some authors (see Gutiérrez et al. (1996) and the references therein) about the existence of multiple optimal designs in many of these optimization problems. Hence, a set of near optimal solutions may contain significantly different designs, where one could eventually identify a robust solution having a good performance under different sets of data. Therefore, a reasonable strategy (Aissi et al. (2009); Averbakh and Lebedev (2004); Kouvelis and Yu (1997); Pan and Nagi (2010); Peng et al. (2011); Rosenhead et al. (1972)) for finding a robust network design could be to take a nearly optimal solution for a variety of future operating scenarios responding to changes in the behavior of the demand, costs or even rare events, instead of seeking for an optimal solution under a fixed set of data that may never happen. Therefore, the minmax regret paradigm may be ideal to deal with the problem. We will adopt the *minmax regret* paradigm with a polyhedral set of possible costs and/or a finite set of demand scenarios.

This minmax regret approach has been used in different applications in the context of designing networks. Peng et al. (2011), used the relative robustness criterion to find reliable logistics networks design with facility disruptions. Specifically, they proposed a design of the network with minimal cost under a set of *nominal* data (the costs when no disruptions occur) while it also performs relatively well when disruptions strike. This last feature is modelled by upper bounding the cost in terms of its relative regret by a pre-specified constant p (*level of robustness*) when disruptions occur. The authors proposed an approach to find efficient solutions under a finite set of data scenarios modelling the possible disruptions. A similar robust optimization model has been recently analized in Fereiduni and Shahanaghi (2017) to design an efficient response immediately following a natural disaster. In this case, the disruptions are caused by natural disasters, such as earthquakes, where, as suggested in the first chapter, it is vital the design of a robust network for humanitarian logistics which will assist in the coordination of a response action.

Pan and Nagi, Pan and Nagi (2010), also included a worst-case absolute regret analysis in their approach to find a robust supply chain design under uncertain demand. They studied this problem into the context of *agile manufacturing*, a concept used to describe new challenges for companies operating in competitive environments in which market opportunities emerge and disappear continually without known patterns. The source of uncertainty in the data is, in this case, not necessarily linked to the lack of valid information to estimate future costs or demand but to the fact that these estimations become quickly obsolete due to the fast changes of the conditions in which the network should operate. The application of robust models seems to be appropriate in such competitive environments in which the need for higher production efficiency and lower operational costs are forcing companies to search for

innovative ways to do business. For instance, some authors have detected new high-risk practices assumed by companies due to the need of purchase from cheaper but less-reliable or unproven suppliers. These policies include new sources of uncertainty in the data as mentioned by Jabbarzadeh et al. (2017). In their paper, these authors provided a robustness approach, named *elastic p-Robustness*, in which the maximum relative regret on costs is used to show a real application of these techniques. Other real applications of a robust optimization model allowing flow bifurcations and based on a set of realizable scenario of the input data could be find in the paper by Lee et al. (2013).

There exist in the literature enough evidences of the computational difficulty of the robust network design both in its theoretical facet and in practical applications. For the particular case of Network Optimization, Minoux (2010) provided a formal $\mathcal{NP}$-hardness proof of a wide family of robust network optimization problems under polyhedral demand uncertainty. In his paper, the author showed that the fact of considering polyhedral demand uncertainty switches from $\mathcal{P}$ to $\mathcal{NP}$-hard the complexity status of one of the considered optimization models. If the uncertainty is considered in the cost data the computational complexity has the same features since these models include, in particular, robust counterparts of flow problems which are known to have $\mathcal{NP}$-hard complexity (see Averbakh and Lebedev (2004); Kouvelis and Yu (1997)). On the other hand, real applications of design network models are difficult to solve even in their deterministic versions due to either large problem sizes and/or nonlinearity of the models (see e.g. Fahimnia et al. (2013)).

The formulation of the deterministic counterpart of a network design model frequently drives us to a Mixed Integer Linear Programming (MILP) problem. Many families of valid inequalities have been derived by exploiting the polyhedral properties of the resulting MILP formulation. These cuts can be used in specialized branch and bound schemes (Atamtürk (2002); Atamtürk and Rajan (2002); Barahona (1996); Günlük (1999)). However, the methodology that may be used more frequently in the context of network design is the one based on a Benders decomposition procedure (see Costa (2005) and the references therein). The nature of these models allows us to decompose the set of decision variables into two sets modelling respectively the topological structure of the network and the flows needed to cover the demands. This fact makes the Benders decomposition method a natural approach since we can use these two sets of variables to define the so-called master and primal problems. Logically, this methodology has been extended to robust network design problems (see Lee et al. (2013) and the references therein). For instance, in one of the first published papers on robust design network, Gutiérrez et al. (1996) proposed a Benders decomposition algorithm to solve a minmax regret problem with a finite set of cost scenarios. Given a tentative structure of the network, they used linear

duality to construct cuts under each scenario. Using these cuts, the algorithm solved simultaneously all the deterministic counterparts of the design problem under the finite set of scenarios, generalizing the existing Benders decomposition algorithm for the deterministic version of the problem. More recently, the same methodology has been used (Lee et al. (2013)) to design a real-life robust telecommunication network under uncertain demand. In this case, the set of demand scenarios is modeled as any vector of a hypercube verifying an additional constraint whose goal is to control the maximum degree of allowed uncertainty.

We will also apply the Benders decomposition methodology to the optimization problems proposed in the following sections of this chapter. Our contribution is to model the uncertainty in data through general polyhedral, and handle it via the minmax regret criterion. This will allow us to describe dependency relations that may exist between the unknown parameters. We will also study the design of a robust network under simultaneous uncertainty in the cost structure and in the demand. In our case, the uncertainty in the demands may also affect to the origin and/or destination of the commodities.

In the following section, a general model for the robust network design is stated using a minmax regret objective function. In Section 2.3.1 the model with uncertain fixed costs (construction, maintenance,...) and uncertain transportation costs is considered. This model is studied together with uncertainty in the origin-destination pairs for the commodities in Section 2.3.2. In the next Section 2.3.3, the model is addressed assuming that uncertainty affects also to the demand and the capacity of the the arcs. Finally, a computational study for the different models is conducted in order to check the scope of the proposed algorithms.

2.2 Model description and deterministic case formulation

We consider

- $N = \{1, \ldots, n\}$ a set of nodes,
- $A \subseteq N \times N$ a set of potential arcs,
- $P \subseteq N \times N$ a list of pairs of *linked* nodes representing the origins and destinations of a set of paths that must be defined in the network,
- f_{ij}^s for all $(i,j) \in A$ the fixed non-negative cost corresponding to the construction of the potential arc (i,j) under a given scenario s and
- c_{ij}^s for all $(i,j) \in A$ the non-negative unitary transportation arc cost under the scenario s of using the potential arc (i,j) to connect directly origin i and destination j. These costs can model any resource that must be spent in order

to send a unit of the considered commodity through the network like fuel, electricity or maintenance operations.

We want to find the subset of potential arcs $A_y \subseteq A$ that should be chosen such that the pairs of origin-destination nodes of the preestablished list P can be connected optimally in the network (N, A_y).

In the following *Mathematical Programming* formulation we will use two subset of decision variables,

- $y_{ij} \in \{0,1\}$ for all $(i,j) \in A$, the *design variables* which determine the subset of potential arcs $A_y \subseteq A$ of the network and
- $x_{ijp} \in \{0,1\}$ for all $(i,j) \in A$, the *flow variables* defining the route chosen to connect the pair $p \in P$ of the list of linked nodes.

A MILP formulation for what we will call hereafter the *Deterministic Network Design* (DND) problem is the following one

$$
\begin{aligned}
Z^*(x) := \min \quad & \sum_{(i,j)\in A} f_{ij}^s y_{ij} + \sum_{(i,j)\in A, p\in P} c_{ij}^s x_{ijp} \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} x_{ijp} - \sum_{(k,i)\in A} x_{kip} = b_{ip}, && i \in N, p \in P, \\
& x_{ijp} \leq y_{ij}, && (i,j) \in A, p \in P, \\
& y_{ij} \in \{0,1\}, && (i,j) \in A, \\
& x_{ijp} \in \{0,1\}, && (i,j) \in A, p \in P,
\end{aligned}
\tag{DND}
$$

where $b_{ip} = 1, b_{jp} = -1$ if $p = (i,j) \in N \times N$ and $b_{ip} = 0$ otherwise.

Remark 1. *In the block of flow constraints of the formulation (DND)*

$$\sum_{(i,j)\in A} x_{ijp} - \sum_{(k,i)\in A} x_{kip} = b_{ip}, \quad i \in N, p \in P,$$

it is implicitly assumed that, if $b_{ip} = 1$ the left hand side of the equation reduces to the first summation and in the case in which $b_{ip} = -1$ it reduces to the second summation. In any case, since all the costs c_{ij}^s are considered nonnegative, any optimal solution will satisfy this requirement because this solution will be free of circular flows. This will apply in any of the following formulations in this chapter thus, for the sake of clarity, we will maintain the compact format of this block of constraints.

□

Remark 2. *Formulation (DND) can be stated in a more general format. For instance, we can restrict the set of possible designs by including limited resources affecting the choice of the subset of arcs (the y-variables). On the other hand, with respect to the x-variables, one can consider more general coefficients* $b_{ip} = b_p, b_{jp} = -b_p$ *for* $p = (i, j) \in P$ *and* $b_{kp} = 0$ *otherwise, being* $b_p \in \mathbb{Z}^+$*. In this case,* b_p *can model the amount of the commodity transported from* i *to* j *or a weight of the relative importance of the pair* $p \in P$ *in comparison with the other pairs. However, for the sake of simplicity, while this element does not play a relevant aspect for the optimization model we will maintain all the* b_p *weights equal to one.*

□

Remark 3. *The binary constraints on the variables* x_{ijp} *can be relaxed to nonnegativity,* $x_{ijp} \geq 0$ *for all* $(i, j) \in A, p \in P$ *in the formulation (DND), due to the separability of its objective function respect to the x-variables and y-variables. Note that the unimodular property can be applied to the inner problem on the x-variables.*

□

We will end this section setting out a simple example of the (DND) problem which will be used to illustrate later formulations, results and behaviors of the solutions.

Example *Grid Design*
Consider the square grid network illustrated in Figure 2.1: a regular support of points (reticle), with 100 nodes, in which from each node arises (if possible) three potential directed arcs, one downward to the node below, one forward to the node on the right, and one diagonally to the node on the right and above. This type of mesh or topology, originally proposed in Cherkassky et al. (1996) and used in Fernández et al. (2014) among others, located on a geography, can model for example pipelines networks or simple circuits.

The circles in the figure represent the nodes, the gray circles represent the origins (suppliers), and the numbers inside them tag each origin; the squares represent destinations (demand points), and the numbers inside the squares, the origin from which each demand point should be supplied. The dashed gray arrows represent potential arcs. All the fixed and unitary transportation arc costs were assumed to be 1 and 0.3 respectively. The problem consists of choosing the network design that connects all the origin-destination nodes at minimum cost.

The arrows highlighted in black color in Figure 2.1 represent an optimal network design after solving the formulation (DND). Note that, since no bounds on the arc capacities are considered, the solution has a very simple design in which several paths overlap. In fact, the optimal network designs under these conditions are frequently spanning trees, arborescences or spanning forests with a few spanning trees.

2.3 Robust models and Benders resolution algorithms

2.3.1 Uncertain costs

In this section a minmax regret version of the (DND) problem is considered under a polyhedral set of cost scenarios. Let S denote the set of possible scenarios which is identified with a polyhedron of costs (c^s, f^s), $s \in S$ whose components are c^s_{ij}, f^s_{ij} for all $(i,j) \in A$. The regret $R(x,y,s)$ of a given solution (x,y) under the scenario $s \in S$ is defined for this problem as the following positive difference

$$R(x,y,s) := \sum_{(i,j)\in A} f^s_{ij} y_{ij} + \sum_{(i,j)\in A, p\in P} c^s_{ij} x_{ijp} - Z^*(s),$$

where $Z^*(s)$ is the optimal value of the (DND) problem under the scenario $s \in S$.

For a given design solution (x,y) it makes sense to compute $R(x,y)$, the maximum regret over the set of possible scenarios S. This value represents the opportunity loss cost for the solution (x,y) and can be obtained by solving the following optimization problem

$$\begin{aligned}
R(x,y) := \quad \max \quad & \sum_{(i,j)\in A} f^s_{ij} y_{ij} + \sum_{(i,j)\in A, p\in P} c^s_{ij} x_{ijp} - \\
& - \sum_{(i,j)\in A} f^s_{ij} \overline{y}_{ij} - \sum_{(i,j)\in A, p\in P} c^s_{ij} \overline{x}_{ijp} \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} \overline{x}_{ijp} - \sum_{(k,i)\in A} \overline{x}_{kip} = b_{ip}, \quad i \in N, p \in P, \qquad (2.1)\\
& \overline{x}_{ijp} \le \overline{y}_{ij}, \quad (i,j) \in A, p \in P, \\
& \overline{y}_{ij} \in \{0,1\}, \quad (i,j) \in A, \\
& \overline{x}_{ijp} \in \{0,1\}, \quad (i,j) \in A, p \in P, \\
& s \in S.
\end{aligned}$$

In order to obtain a complete formulation of the problem (2.1) we will assume that a polyhedral description of the set of fixed and unitary transportation arc costs is known,

$$S := \left\{ (c,f) \ge 0 : \sum_{(i,j)\in A} (a^f_{ijk} f_{ij} + a^c_{ijk} c_{ij}) \le d_k,\ k \in K \right\},$$

where we have used the same letter S that already identified the set of scenarios to make the notation simpler. Additionally, it will be assumed the compactness of this polyhedron of costs, thus there are constants $\overline{c}$ and $\overline{f}$ such as $0 \le c_{ij} \le \overline{c}$ and $0 \le f_{ij} \le \overline{f}$ for all $(i,j) \in A$.

Using this polyhedral representation of the set S of possible cost scenarios, a

formulation of the optimization problem (2.1) can be given by

$$
\begin{aligned}
R(x,y) := \quad \max \quad & \sum_{(i,j)\in A} f_{ij}y_{ij} + \sum_{(i,j)\in A, p\in P} c_{ij}x_{ijp} - \\
& - \sum_{(i,j)\in A} f_{ij}\overline{y}_{ij} - \sum_{(i,j)\in A, p\in P} c_{ij}\overline{x}_{ijp} \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} \overline{x}_{ijp} - \sum_{(k,i)\in A} \overline{x}_{kip} = b_{ip}, \quad i \in N, p \in P, \\
& \overline{x}_{ijp} \leq \overline{y}_{ij}, \quad (i,j) \in A, p \in P, \\
& \sum_{(i,j)\in A} (a^f_{ijk} f_{ij} + a^c_{ijk} c_{ij}) \leq d_k, \; k \in K, \\
& \overline{y}_{ij} \in \{0,1\}, \quad (i,j) \in A, \\
& \overline{x}_{ijp} \in \{0,1\}, \quad (i,j) \in A, p \in P, \\
& \overline{f}_{ij} \geq 0, \quad (i,j) \in A, \\
& \overline{c}_{ij} \geq 0, \quad (i,j) \in A.
\end{aligned}
\tag{2.2}
$$

The above formulation contains quadratic terms in the objective function which added to the binary nature of some of the variables increase the difficulty to solve the problem. Nonetheless, as stated in the following result, this nonlinear mixed integer problem is equivalent to the following MILP formulation

$$
\begin{aligned}
R(x,y) := \quad \max \quad & \sum_{(i,j)\in A} f_{ij}y_{ij} + \sum_{(i,j)\in A, p\in P} c_{ij}x_{ijp} - \\
& - \sum_{(i,j)\in A} w_{ij} - \sum_{(i,j)\in A, p\in P} z_{ijp} \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} \overline{x}_{ijp} - \sum_{(k,i)\in A} \overline{x}_{kip} = b_{ip}, \quad i \in N, p \in P, \\
& \overline{x}_{ijp} \leq \overline{y}_{ij}, \quad (i,j) \in A, p \in P, \\
& \sum_{(i,j)\in A} (a^f_{ijk} f_{ij} + a^c_{ijk} c_{ij}) \leq d_k, \; k \in K, \\
& c_{ij} - \overline{c}(1 - \overline{x}_{ijp}) \leq z_{ijp}, \quad (i,j) \in A, p \in P, \\
& 0 \leq z_{ijp}, \quad (i,j) \in A, p \in P, \\
& f_{ij} - \overline{f}(1 - \overline{y}_{ij}) \leq w_{ij}, \quad (i,j) \in A, \\
& 0 \leq w_{ij}, \quad (i,j) \in A, \\
& \overline{y}_{ij} \in \{0,1\}, \quad (i,j) \in A, \\
& \overline{x}_{ijp} \in \{0,1\}, \quad (i,j) \in A, p \in P, \\
& f_{ij} \geq 0, \quad (i,j) \in A, \\
& c_{ij} \geq 0, \quad (i,j) \in A.
\end{aligned}
\tag{PP-RND}
$$

Proposition 5. *The optimization problem* (2.1) *is equivalent to the MILP formulation (PP-RND).*

Proof. The blocks of constraints of the (PP-RND) formulation

$$\begin{aligned} &c_{ij} - \overline{c}(1 - \overline{x}_{ijp}) \leq z_{ijp}, \quad (i,j) \in A, p \in P, \\ &0 \leq z_{ijp}, \quad (i,j) \in A, p \in P, \end{aligned}$$

imply that $z_{ijp} = \max\{0, c_{ij} - \overline{c}(1 - \overline{x}_{ijp})\}$ since z_{ijp} appears with a minus sign in the objective function that must be maximized. Taking into account that the constant $\overline{c}$ is an upper bound on the value of the possible unitary transportation arc cost, that is, $c_{ij} - \overline{c} \leq 0$ for all $(i,j) \in A$, one has that

$$\max\{0, c_{ij} - \overline{c}(1 - \overline{x}_{ijp})\} = c_{ij} x_{ijp}, \quad (i,j) \in A, p \in P.$$

The same argument applies for the w_{ij} variables for all $(i,j) \in A$. □

Remark 4. *The binary constraints on the variables $\overline{x}_{ijp}$ cannot, in general, be removed from the (PP-RND) formulation, contrary to what was mentioned in Remark 3 referent to the (DND) formulation. In this case, the separability reasoning of that remark drives us to obtain an inner optimization problem in which a convex piecewise linear function must be minimized. Since its optima need not be an extreme point of the relaxed polyhedron of feasibility, the unimodular property cannot be applied.*

□

Formulation (PP-RND) allows us to assess the opportunity loss of a given design according to the above polyhedral model for the uncertainty on the cost structure. Hence, if we need to compare a finite (and small) number of network designs it suffices to choose one of them reaching the minimum of these assessments. However, we are interested here in the problem of finding a robust network design when the number of possible choices is too large to assess the opportunity loss cost of each one of them. If we consider all the possible choices of subsets of arcs A_y making possible to connect the pairs of P, the resulting problem to be solved would be the following one

$$\begin{aligned} R^* := \min \quad & R(x,y) \\ \text{subject to:} \quad & \\ & \sum_{(i,j)\in A} x_{ijp} - \sum_{(k,i)\in A} x_{kip} = b_{ip}, \quad i \in N, p \in P, \\ & x_{ijp} \leq y_{ij}, \quad (i,j) \in A, p \in P, \\ & y_{ij} \in \{0,1\}, \quad (i,j) \in A, \\ & x_{ijp} \in \{0,1\}, \quad (i,j) \in A, p \in P. \end{aligned} \tag{RND}$$

Note that, we have just replaced the objective function of the deterministic model (DND) by the opportunity loss cost in (RND). This change represents a substantial change in the resolution methodology since $R(x, y)$ is, in general, a nonlinear function for which the MILP formulation (PP-RND) must be solved to assess it in a given point.

In order to find a robust design we will solve the problem (RND) by a Benders decomposition algorithm. It starts with a given design (x^0, y^0) and assess its maximum regret by solving the problem (PP-RND) which will be the *primal problem*. Its optimal objective value represents an upper bound on the optimal value of (RND) and one of its optimal solutions defines a *worst-case adversary*, $(\overline{x}^0, \overline{y}^0)$, and a *worst-case scenario*, $s^0 := (c^0, f^0)$, for the initial design (x^0, y^0). In each iteration, we will find a minmax regret design for the finite set of worst-case scenarios of the designs previously assessed. The minmax regret value of this last solution will be assessed by solving its corresponding primal problem (PP-RND) and the process continues.

The new design found in each iteration is a minmax regret solution for the subset of considered scenarios of costs. The corresponding optimization problem is the *master problem* and its optimum is a lower bound on the optimal objective value of the problem (RND). For a generic iteration n, the master problem has the following formulation

$$
\begin{aligned}
r^* := \min \quad & r \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} f_{ij}^q y_{ij} + \sum_{(i,j)\in A, p\in P} c_{ij}^q x_{ijp} - R(\overline{x}^q, \overline{y}^q, s^q) \leq r, q = 0, ..., n-1, \\
& \sum_{(i,j)\in A} x_{ijp} - \sum_{(k,i)\in A} x_{kip} = b_{ip}, i \in N, p \in P, \\
& x_{ijp} \leq y_{ij}, (i,j) \in A, p \in P, \\
& y_{ij} \in \{0,1\}, (i,j) \in A, \\
& x_{ijp} \in \{0,1\}, (i,j) \in A, p \in P.
\end{aligned}
\tag{MP-RND}
$$

Hence, the resolution of the primal and master problems in each iteration allows us to update a pair of lower and upper bounds on the optimal objective of the problem (RND) whose positive difference is used to stop the procedure which is stated now as Algorithm 1.

Proposition 6. *If $\varepsilon = 0$, Algorithm 1 ends in a finite number of iterations with an optimal solution of the minmax regret network design problem (RND).*

Proof. First observe that we can assume that the list L of worst-case scenarios can be included in the finite set of vertices of the polyhedron of scenarios of costs S. This

Algorithm 1

1: **procedure** INITIALIZATION
2: Let $(x^\varepsilon, y^\varepsilon) := (x^0, y^0)$ be an initial feasible design.
3: Solve the primal problem (PP-RND) to obtain the opportunity loss cost $R(x^0, y^0)$.
4: Let $s^0 := (c^0, f^0)$ be a worst-case cost scenario for (x^0, y^0).
5: Initialize the list $L := \{(c^0, f^0)\}$.
6: Set $\beta := R(x^0, y^0)$, an upper bound on the optimal value of (RND).
7: Go to the iteration $n := 1$.
8: **procedure** ITERATION $(n = 1, 2, \ldots)$
9: Solve the master problem (MP-RND) and let (x^n, y^n) be one of its optimal solutions.
10: Set $\alpha := r^*$ the optimal value of the problem (MP-RND).
11: **if** $\beta - \alpha < \varepsilon$ **then**
12: $(x^\varepsilon, y^\varepsilon)$ **is an ε-approximation to the problem (RND).**
13: **else**
14: Solve the primal problem (PP-RND) to obtain the value $R(x^n, y^n)$.
15: Let $s^n := (c^n, f^n)$ be a worst-case cost scenario for (x^n, y^n) and $(\overline{x}^n, \overline{y}^n)$ a worst-case adversary.
16: Update the list $L := L \cup \{(c^n, f^n)\}$.
17: **if** $R(x^n, y^n) < \beta$ **then**
18: Update the upper bound $\beta := R(x^n, y^n)$ and $(x^\varepsilon, y^\varepsilon) := (x^n, y^n)$
19: Go to the iteration $n := n + 1$.

holds since, by (2.2), the maximum regret $R(x, y)$ of any network design is attained as the maximum of a convex piecewise linear function on a bounded polyhedron.

In order to prove the finiteness of Algorithm 1 we will show how the cardinality of the list L increases in one unit in each iteration. Since we have seen the set of possible worst-case scenarios can be considered finite, the number of iterations must be finite too.

On the contrary, if the cardinality of L remains unchanged after an iteration n, the worst-case scenario found in Line 15 of Algorithm 1 must already belong to the list L, that is, this scenario has been generated in a previous iteration. In this case,

$$R(x^n, y^n) = R(x^n, y^n, s^n) = \max_{q=0,1,\ldots,n-1} \{R(x^n, y^n, s^q)\} = \alpha \leq R^* \leq R(x^n, y^n),$$

where R^* is the optimal value of the problem (RND) and it has been used that (x^n, y^n) is an optimal solution of (MP-RND) for the list of scenarios $\{s^q, q = 0, 1, \ldots, n-1\}$ (see Line 9). Since in the above chain of inequalities, both extreme are equal we have that Algorithm 1 will end in the following iteration $n+1$ with the optimal solution (x^n, y^n). □

Example *Grid Design* (continuation)
In the first part of this example we assumed that the fixed and unitary transportation arc costs were known. Now we will consider that some arc costs are affected by uncertainty which can be caused by different situations. Let us think, for instance, that we are designing a transportation network in which we are considering to dig a tunnel through a mountain. Its construction cost can be very variable depending, for instance, on whether unexpected materials making more difficult the digging tasks are found or not. In other projects, the source of the uncertainty can be the lack of experience making some of the involved construction tasks. For example, in the *Haramain High-Speed Railway* recently opened to the public in Saudi Arabia, connecting the Muslim holy cities of Medina and Mecca, the little experience in the construction of rails over sand was one of the main sources of uncertainty about the construction and maintenance costs (in economical and construction length terms).

In order to illustrate such features in our toy example we will identify by a rectangular a *unsure zone*, that is, a set of potential arcs whose fixed costs may vary in an uncertain amount. In Figure 2.2 this unsure zone has been depicted by a grey rectangle where the enclosed arcs are those affected by the uncertain costs. We assume that the fixed costs for all those arcs can now vary, increasing or decreasing its value around the previously fixed cost of 1 unit, holding this value as the mid-point of the interval of variation, $[0.4, 1.6]$.

Figure 2.2 shows the network design that minimizes the maximum regret under this new cost structure, after solving the RND problem with Algorithm 1. We can observe that the network structure has changed significantly in order to avoid the construction of the potential arcs with uncertain fixed costs. Then, incorporating uncertain costs to a road design may significantly change the structure of the solution by trying to dodge, if possible, the construction of the potential arcs having uncertain costs due to the presence of the unsure zone. This change in the network design seems to be a reasonable (robust) answer to the new unpredictable cost structure and shows the sensitiveness of our model to the uncertain costs.

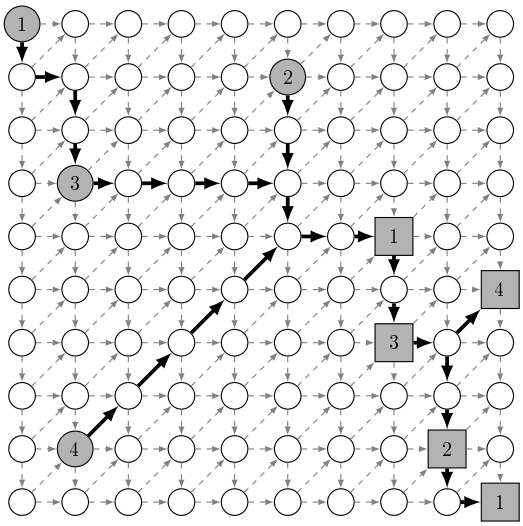

Figure 2.1: Example of (DND).

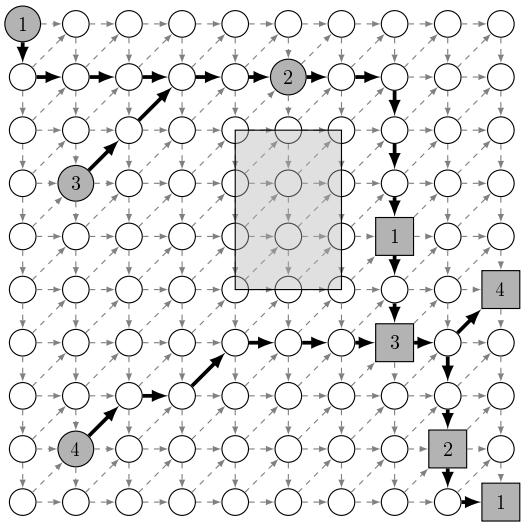

Figure 2.2: A robust design avoiding the construction of arcs with high cost uncertainty.

2.3.2 Uncertain costs and linked pairs

In this section we will study how to find a robust design of the network when the list of linked pairs is partially known. In this case, P becomes the list of pairs that *could* be linked, that is, the set of pairs that eventually could be connected under a possible scenario, but not necessarily all of them will be connected in a realized scenario. Hence, each scenario will define the subset of these pairs that are effectively connected and the corresponding transportation costs (c_{ij}^s). In this new context, each feasible design will be defined by a set of arcs A_y, identified through the binary variables y as in the previous model, and a set of flow variables x that must determine how to connect any pair of origin-destination nodes of P independently of whether they must be connected or not under the scenario that finally occurs. That is, since the scenario that finally will occur is not known a priori, the robust solution must include how to connect each pair of P, just in case the connection is given.

As discussed in the introduction, the set of possible scenarios of connected pairs may model potential business relations or flow patterns among nodes requiring some type of connection through the network and verifying a given set of technical requirements. To identify these pairs a new set of binary variables $u_p, p \in P$ is defined and it will be assumed that all the technical requirements can be modeled using the following constraints

$$\sum_{p \in P} e_{pr} u_p \leq g_r, \quad r \in R,\ u_p \in \{0,1\}, \forall p \in P. \tag{2.3}$$

For instance, if $R = \{1\}$ and $e_{p1} = 1$ for all $p \in P$, any scenario in which at most g_1 linkages in P may happen.

Hence, for a given design of the network, a worst-case scenario will detect a set of pairs to be linked and a structure of costs such that, we can find a new design (its worst-case adversary design) connecting those pairs of nodes as cheap as possible, in comparison with the cost incurred by the initially considered design. In order to find such a worst-case scenario, we will extend the formulation (PP-RND) to this new context. To do this we will integrate in the formulation the new constraints modelling the scenarios of pairs of nodes to be linked as it is shown in (PPUL-RND).

$$
\begin{aligned}
R(x,y) := \quad \max \quad & \sum_{(i,j)\in A} f_{ij} y_{ij} + \sum_{(i,j)\in A, p\in P} \xi_{ijp} x_{ijp} - \\
& - \sum_{(i,j)\in A} w_{ij} - \sum_{(i,j)\in A, p\in P} z_{ijp} \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} \overline{x}_{ijp} - \sum_{(k,i)\in A} \overline{x}_{kip} = b_{ip}, \quad i \in N, p \in P, \\
& \overline{x}_{ijp} \leq \overline{y}_{ij}, \quad (i,j) \in A, p \in P, \\
& \sum_{(i,j)\in A} (a^f_{ijk} f_{ij} + a^c_{ijk} c_{ij}) \leq d_k, \; k \in K, \\
& \sum_{p\in P} e_{pr} \overline{u}_p \leq g_r, \; r \in R, \\
& c_{ij} - \overline{c}(2 - \overline{u}_p - \overline{x}_{ijp}) \leq z_{ijp}, \quad (i,j) \in A, p \in P, \\
& 0 \leq z_{ijp}, \quad (i,j) \in A, p \in P, \\
& \xi_{ijp} \leq c_{ij}, \quad (i,j) \in A, p \in P, \\
& 0 \leq \xi_{ijp} \leq \overline{c}\, \overline{u}_p, \quad (i,j) \in A, p \in P, \\
& f_{ij} - \overline{f}(1 - \overline{y}_{ij}) \leq w_{ij}, \quad (i,j) \in A, \\
& 0 \leq w_{ij}, \quad (i,j) \in A, \\
& \overline{y}_{ij} \in \{0,1\}, \quad (i,j) \in A, \\
& \overline{x}_{ijp} \in \{0,1\}, \quad (i,j) \in A, p \in P, \\
& \overline{u}_p \in \{0,1\}, \quad p \in P, \\
& f_{ij} \geq 0, \quad (i,j) \in A, \\
& c_{ij} \geq 0, \quad (i,j) \in A.
\end{aligned}
\tag{PPUL-RND}
$$

Now, for a given design (x, y), the formulation (PPUL-RND) will find the design of a worst-case adversary network $(\overline{x}, \overline{y})$, a possible structure of fixed costs, f, unitary transportation arc costs, c, and a subset of origin-destination nodes, $\overline{u}$, whose linkage is favoured under this scenario of costs compared with the linkage cost needed in the original design.

In the formulation (PPUL-RND) the continuous variables ξ_{ijp} for all $(i,j) \in A$

and $p \in P$ have been included to identify if the connection cost of the pair p needs to be added, or not, to the overall cost of the design (x, y). Observe that the constrains

$$\begin{aligned} \xi_{ijp} &\leq c_{ij}, \quad (i,j) \in A, p \in P, \\ 0 \leq \xi_{ijp} &\leq \overline{c}\,\overline{u}_p, \quad (i,j) \in A, p \in P, \end{aligned}$$

ensure that ξ_{ijp} will only take the unitary transportation arc cost c_{ij} when the binary variable $\overline{u}_p$ is equal to one, that is, when the pair of potentially linked nodes pointed by p is included in the set of effectively linked nodes under the considered scenario. Otherwise, these variables are zero for every $(i,j) \in A$.

Remark 5. *Observe that, the equations*

$$\sum_{(i,j)\in A} \overline{x}_{ijp} - \sum_{(k,i)\in A} \overline{x}_{kip} = b_{ip}, \quad i \in N\, p \in P,$$

of the formulation (PPUL-RND) make that all the feasible solutions considered as possible worst-case designs for the solution (x, y) must allow the connection of every potential pair of nodes $p \in P$. In other case, if we compare the solution (x, y) with adversary designs in which it were only guaranteed the effective connections of those pairs appearing in each assessed scenario we would be inserting an asymmetry into the optimization model (PPUL-RND) since the solution (x, y) is required to have the availability of connecting every potential pair on nodes of P. This asymmetry would give rise to positive regrets even for those solutions (x, y) connecting optimally all the possible pairs under every possible scenario of costs due to the fact that its worst case adversary may not require the construction of the arcs which would assure all the possible connections.

Hence in order to assess the maximum regret of a given solution using the formulation (PPUL-RND), the fixed costs due to the choice of designs guaranteeing all the potential connections are taken into account both in the considered solution as in its worst-case adversary. However, in both cases the connecting costs are only considered among those pairs of P appearing in the worst-case scenario. This fact is modeled through the constraints affecting to the variables ξ_{ijp}, as it was commented earlier, together with the constraints lower bounding the variables z_{ijp},

$$c_{ij} - \overline{c}(1 - \overline{u}_p) \leq z_{ijp}, \quad (i,j) \in A, p \in P.$$

□

Hence, for any given solution (x, y), the *primal problem* (PPUL-RND) finds a worst-case design for it and a worst-case scenario of costs together with the pairs that must be linked in order to reach its maximum regret. Algorithm 1 can be modified

to solve this new optimization model in accordance with a new MILP formulation of the master problem that now becomes

$$
\begin{aligned}
&\min \quad r \\
&\text{subject to:} \\
&\sum_{(i,j)\in A} f_{ij}^q y_{ij} + \sum_{(i,j)\in A, p\in P} c_{ij}^q \overline{u}_p^q x_{ijp} - R(\overline{x}^q, \overline{y}^q, s^q) \leq r, q = 0, 1, ..., n-1, \\
&\sum_{(i,j)\in A} x_{ijp} - \sum_{(k,i)\in A} x_{kip} = b_{ip}, i \in N, p \in P, \\
&x_{ijp} \leq y_{ij}, (i,j) \in A, p \in P, \\
&y_{ij} \in \{0,1\}, (i,j) \in A, \\
&x_{ijp} \in \{0,1\}, (i,j) \in A, p \in P,
\end{aligned}
\tag{MPUL-RND}
$$

In the formulation (MPUL-RND) we must observe the following,

- for each list of worst-case scenarios for the linked pairs, given by the binary vector $\overline{u}^q$, $q = 0, 1, \ldots, n-1$, the first block of constraints compares the overall (construction and connection) cost under the considered design with the optimum under the corresponding scenarios of costs and linked pairs,
- the second block of constraints defines a feasible set of flow variables allowing to connect every pair of potentially linked nodes.

Proposition 7. *The optimal value of the problem (MPUL-RND) is a lower bound on the optimal objective of the problem (RND) under uncertain structure of costs and list of linked pairs.*

Proof. The minimum value of r for each feasible design of the network in the problem (MPUL-RND) represents the maximum regret of the considered design under the finite set of worst-case scenarios that defines this problem. □

We will now state the Benders decomposition algorithm associated to the new formulations of the primal and master problems which is a slight modification of Algorithm 1.

Proposition 8. *If $\varepsilon = 0$, Algorithm 2 ends in a finite number of iterations with an optimal solution of the minmax regret network design problem (RND) under an uncertainty set of linked pairs of nodes.*

Proof. Given a feasible design of the network (x, y) and a feasible subset of linked pairs u, the formulation (PPUL-RND) ensures that one can find an extreme point of the bounded polyhedron of possible cost scenarios S attaining the maximum regret. Since the feasible set of subsets of linked pairs u is finite we can assume the list

Algorithm 2

procedure INITIALIZATION
  Let $(x^\varepsilon, y^\varepsilon) := (x^0, y^0)$ be an initial feasible design.
  Solve the primal problem (PPUL-RND) to obtain the opportunity loss cost $R(x^0, y^0)$.
  Let $s^0 := (c^0, f^0, u^0)$ be a worst-case scenario for (x^0, y^0).
  Initialize the list $L := \{(c^0, f^0, u^0)\}$.
  Set $\beta := R(x^0, y^0)$, an upper bound on the optimal value of (RND) with uncertain linked nodes.
  Go to the iteration $n := 1$.
procedure ITERATION $(n = 1, 2, \ldots)$
  Solve the master problem (MPUL-RND) and let (x^n, y^n) be one of its optimal solutions.
  Set $\alpha := r^*$ the optimal value of the problem (MPUL-RND).
  if $\beta - \alpha < \varepsilon$ **then**
    $(x^\varepsilon, y^\varepsilon)$ **is an ε-approximation to the problem (RND) with uncertain linked nodes.**
  else
    Solve the primal problem (PPUL-RND) to obtain the value $R(x^n, y^n)$.
    Let $s^n := (c^n, f^n, u^n)$ be a worst-case scenario for (x^n, y^n) and $(\overline{x}^n, \overline{y}^n, \overline{u}^n)$ a worst-case adversary.
    Update the list $L := L \cup \{(c^n, f^n, u^n)\}$.
    if $R(x^n, y^n) < \beta$ **then**
      Update the upper bound $\beta := R(x^n, y^n)\}$ and $(x^\varepsilon, y^\varepsilon) := (x^n, y^n)$
    Go to the iteration $n := n + 1$.

L of Algorithm 2 is a subset of a finite set of scenarios. Hence following the same reasoning that proof in Proposition 6 one can conclude the finiteness of Algorithm 2 with $\varepsilon = 0$. □

Example *Grid Design* (continuation)
Up until now, we have assumed that five of the pairs of origin-destination nodes had to be linked. Nevertheless, in real life there exist design problems in which the pairs to be linked by the network could be only partially known. As commented in the introduction, in the context of agile manufacturing, companies must operate competently in a continuously changing market, being able to adapt their distribution networks to new potential suppliers. An example of this situation can be found in the field of companies providing Internet access, where the demand is continuously increasing and the suppliers can change due to the improvements carried out in the equipments. With the aim of covering these situations in our illustrative example we will now assume that an uncertain number of pairs from the list P will be linked under any of the possible scenarios. Here we assume than no more than two pairs will be linked under any scenario. However, additional constraints about the possible subset of linked pairs could be considered.

Note that the new conditions increase considerably the existing uncertainty since the design must ensure the eventual linkage of any pair of the list P while its efficiency will be measured confronting this design with the optimal design under any selection of a subset of almost two pairs from the list P to link. The proposed design is sensitive to the new conditions as can be seen by comparing the optimal solutions shown in the figures 2.2 and 2.3. When we solve the new model via Algorithm 2, the optimal design of the network depicted in Figure 2.3 is obtained. In this design, four of its arcs belong to the rectangular unsure zone while the solution of the previous model, shown in Figure 2.2, avoids using arcs from this zone. The explanation to this more *risky* design is that the uncertainty about the unitary transportation costs of the arcs only affects the regret if the worst scenario includes the corresponding arc to join one of the origin-destination pairs whose linkage is mandatory. Otherwise, the cost does not affect the objective function. Assessing the optimal design of the previous model under this new objective function one obtains a value of its maximum regret 5.66% higher than the optimal value.

2.3.3 Uncertain costs, linked pairs and demands

Let us now consider the situation in which unknown demands must be served in certain nodes from specified sources. Specifically, let $p = (i, j) \in P$ be a possible pair of linked nodes, we consider the parameters $b_{ip} = b_p, b_{jp} = -b_p$ and $b_{kp} = 0$ for $k \notin \{i, j\}$ (see Remark 2), where b_p is an unknown amount of certain commodity that

must be served. If we can assume the uncertain values b_p are non-negative integers upper bounded by $\bar{b}_p$ the corresponding minmax regret problem can be modeled using the formulation of Section 2.3.2 in which a set of $\bar{b}_p$ replicas of the pair p would be added to the set of potential linked pairs. Since in the model of the previous subsection each possible scenario defines, in particular, the subset of potential linked nodes that must be managed by the distribution network, we are allowing, in fact, the existence of uncertain flows of commodities among the considered nodes, that is, if under a given scenario one has that b replicas of the pair p occur it means that a demand of b units is served at the node j from the node i, where $p = (i, j)$. A drawback of this approach is the considerable increase in the size of both, primal and master problems, of the corresponding model since each replicated pair p entails a new set of variables x_{ijp} and an additional variable u_p, which makes a total of $|A|+1$ new binary variables. This increase in the number of binary variables implies, for instance, $2|A|$ new continuous variables z_{ijp} and ξ_{ijp} and $4|A|$ new constraints for the primal formulation (PPUL-RND).

However, the idea of using replicas of the linked pairs of nodes can be used with a much less increasing in the size of the involved problems if a finite set of demand configurations is managed. Let us suppose, for instance, that the demand of a given node j from a node i can be just one of the values of a finite set $\{b_p^1, \ldots, b_p^r\}$ including the zero value, where p denotes de pair (i, j). Under this hypothesis, we can consider that all the demand is transported just as an only block by adjusting the unitary transportation arc costs. Doing that, we only need a replica of the flow and its corresponding auxiliary variables for each index in the set $D_p = \{1, \ldots r\}$ for each pair $p \in P$. We can even make smaller the increasing in size of the problems if we consider a finite set of demand configurations that simultaneously fix the demands of all the linked pairs.

In the set of potential demands of a given pair p, the zero value means that this pair should not be connected under this scenario and then its contributions to the objective function in terms of transportation costs is zero. However, even if this null value is the realization of the demand for the given pair, every network design under consideration must allow us to connect this pair of nodes and hence the corresponding contribution in terms of fixed costs, must be added to the objective function. In order to do that we consider the set of binary variables $\{u_{p1}, \ldots, u_{pr}\}$ to model the posible scenario of demands associated to the pair p, and include the constraints

$$\sum_{s \in D_p} u_{ps} = 1,\ p \in P,$$

which will be used, together with the flow variables x, to force the feasible designs to have the availability of connecting every potential pair $p \in P$ of linked nodes. The

goal is to avoid asymmetries as those discussed in the previous section when a design is compared with its worst-case adversary. Finally, one can slightly modify the formulation (PPUL-RND) to accommodate the new conditions in the new formulation (PPUD-RND) using once again the original values $b_{ip} \in \{0, 1, -1\}$, $i \in N, p \in P$ of the (DND) formulation .

$$
\begin{aligned}
R(x,y) := \quad \max \quad & \sum_{(i,j)\in A} f_{ij} y_{ij} + \sum_{(i,j)\in A, p\in P, s\in D_p} \xi_{ijps} x_{ijp} - \\
& - \sum_{(i,j)\in A} w_{ij} - \sum_{(i,j)\in A, p\in P, s\in D_p} z_{ijps} \\
\text{subject to:} \quad & \\
& \sum_{(i,j)\in A} \overline{x}_{ijp} - \sum_{(k,i)\in A} \overline{x}_{kip} = b_{ip}, \quad i \in N, p \in P, \\
& \overline{x}_{ijp} \le \overline{y}_{ij}, \quad (i,j) \in A, p \in P, \\
& \sum_{(i,j)\in A} (a^f_{ijk} f_{ij} + a^c_{ijk} c_{ij}) \le d_k, \; k \in K, \\
& \sum_{s\in D_p} \overline{u}_{ps} = 1, \; p \in P, \\
& \sum_{p\in P, s\in D_p} e_{psr} \overline{u}_{ps} \le g_r, \; r \in R, \\
& c_{ij} b^s_p - \overline{c}\overline{b}(2 - \overline{u}_{ps} - \overline{x}_{ijp}) \le z_{ijps}, \\
& \qquad\qquad (i,j) \in A, p \in P, s \in D_p, \\
& 0 \le z_{ijps}, \quad (i,j) \in A, p \in P, s \in D_p, \\
& \xi_{ijps} \le c_{ij} b^s_p, \quad (i,j) \in A, p \in P, s \in D_p, \\
& 0 \le \xi_{ijps} \le \overline{c}\overline{b}\overline{u}_{ps}, \quad (i,j) \in A, p \in P, s \in D_p, \\
& f_{ij} - \overline{f}(1 - \overline{y}_{ij}) \le w_{ij}, \quad (i,j) \in A, \\
& 0 \le w_{ij}, \quad (i,j) \in A, \\
& \overline{y}_{ij} \in \{0,1\}, \quad (i,j) \in A, \\
& \overline{x}_{ijp} \in \{0,1\}, \quad (i,j) \in A, p \in P, \\
& \overline{u}_{ps} \in \{0,1\}, \quad p \in P, \\
& f_{ij} \ge 0, \quad (i,j) \in A, \\
& c_{ij} \ge 0, \quad (i,j) \in A.
\end{aligned}
$$

(PPUD-RND)

The new formulation (PPUD-RND) contains a large number of mixed variables which will make harder its numerical resolution, however the resulting robust model cover a considerable set of uncertain parameters in the cost-demand structure of the problem. It is also interesting the implications of considering linear relation to model the possible linkages between these parameters. For example, the block of constraints

$$\sum_{p\in P, s\in D_p} e_{psr} u_{ps} \le g_r, \; r \in R,$$

can be used to model how to serve at least a given amount of total demand B among the considered pairs of nodes,

$$\sum_{p\in P, s\in D_p} b_p^s u_{ps} \geq B.$$

To end this section we will briefly consider the case in which the demand of the commodity can be satisfied using a route with bifurcations on the network. These solutions can be interesting in the case in which the arcs have limited capacities. If we consider only a finite set of possible capacities for each arc, the corresponding problem can be modeled using replicas of the design variables. Specifically, if the capacity of the arc (i,j) is a new decision variable of the design, we can use t replicas of the corresponding binary variable y_{ij}, $\{y_{ij1},\ldots,y_{ijt}\}$ and modify the block of constraints relating the design variables, y, and the flow variables, x, in the formulation (PPUD-RND) as follows

$$\sum_{p\in P, s\in D_p} b_p^s \overline{x}_{ijps} \leq \sum_t K_t \overline{y}_{ijt}, \quad (i,j)\in A,$$

and

$$\sum_t \overline{y}_{ijt} = 1, \quad (i,j)\in A,$$

where K_t represents each value of the capacity for the arc $(i,j)\in A$.

Given the primal problem (PPUD-RND) of this general model, a similar procedure of Benders decomposition to that described in Section 2.3.2 can be used to solve the corresponding problem. In the next section, it is carried out an experimental study on the computational time needed to solve some of these formulations. This can serve as an indicator of the *level of applicability* of these models in real situations.

Example *Grid Design* (continuation)
To conclude the numerical example used throughout this chapter, the variant of uncertain demand is considered. Once again, this feature can model actual situations in which the traffic or the commodity flow between origin-destination nodes is only partially known. An example of this situation can be found in the existing traffic in communication networks. In most cases, there exist no precise information about the expected traffic between the terminals to be connected. However, a collection of possible traffic matrices can be available. With the aim of covering this situation, we will assume known a set of possible demands for each of the five pairs that could be eventually linked under a possible scenario. These sets are respectively, $\{0,1,2,3\}$, $\{0,1,2\}$, $\{0,1\}$, $\{0,1\}$ and $\{0,1,2,3\}$ for the ordered set of pairs of P. Any configuration of demands is allowed, that is, we consider that any choice of the demand values from these sets can occur under a possible scenario and we maintain

the constraint referring to the maximum number of two effective linkage under any possible scenario. The values considered for the demands could represent the usual traffic value as 1, twice the usual traffic value as 2 and so on. The value 0 represents the absence of traffic between the pair of nodes. If the constraint of a maximum number of effective linkage is removed the resulting problem should consider all the scenarios corresponding to the effective linkage of the pairs in any subset of the list P. In our case, with the constraint of at most two effective linkages, these values of 0-demand are redundant.

Solving the problem via Benders decomposition, as described in this section, we obtain an optimal design which is shown in Figure 2.4. Since the set of possible scenarios contains all the possible scenarios of the previous considered variant, the optimal regret increases 5.66% the optimal regret of the previous model. Moreover, the optimal design for the example assuming uncertain costs and linked pairs, represented in Figure 2.3, is not optimal in this new variant of (RND) achieving a maximum regret under the new conditions 58.93% higher that the optimal value. This shows, once again, the sensitiveness of the optimal design under variation on the uncertainty sources.

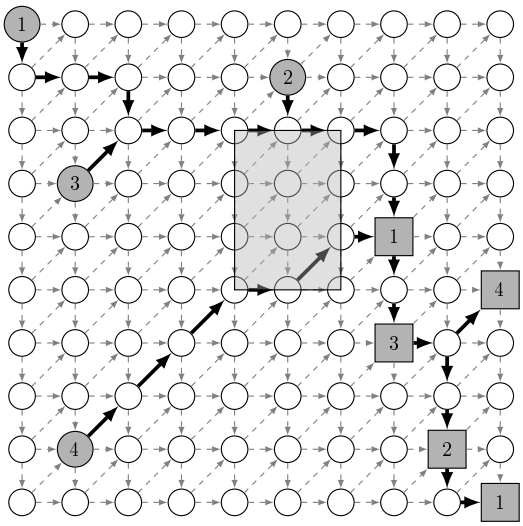

Figure 2.3: Optimal design assuming uncertain costs and linked pairs.

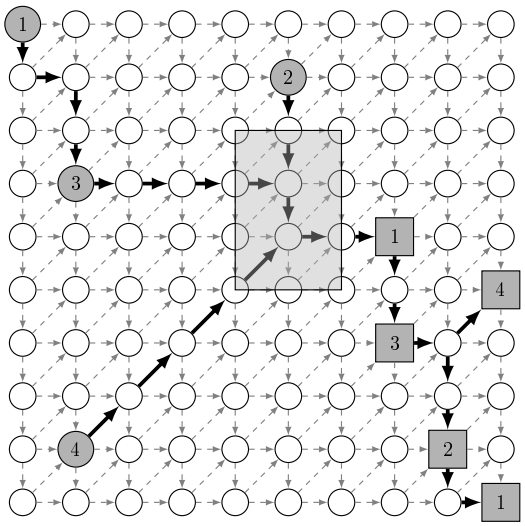

Figure 2.4: Optimal design assuming uncertain costs, linked pairs and demands.

2.4 Computational experiment

In this last section we report some numerical experiments conducted to check the scope and speed of the proposed algorithms and to analyze the structure of the solutions in different instances of the (RND) problem under:

1. uncertain costs,
2. uncertain costs and linked pairs and
3. uncertain costs, linked pairs and demands.

In order to conduct the computational study, we generated instances of directed acyclic square grid networks as the one used in the *Grid Design* example. Specifically, the process defines the following elements:

- A set of $n = m \times m$ nodes corresponding to points on the plane with integer coordinates $[x, y]$, $1 \leq x \leq m$, $1 \leq y \leq m$.
- A set of *forward* arcs of the form $([x, y], [x + 1, y])$, $1 \leq x \leq m - 1$, $1 \leq y \leq m$.
- A set of *downward* arcs of the form $([x, y], [x, y - 1])$, $1 \leq x \leq m$, $2 \leq y \leq m$.
- A set of *diagonal* arcs of the form $([x, y], [x + 1, y + 1])$, $1 \leq x \leq m - 1$, $1 \leq y \leq m - 1$.

The grid structure of our instances could have certain actual applications but what is more important for our experimental study is that the considered topology allows us to control possible infeasibility due to the inclusion of disconnected pairs of nodes in the list P. In our case, nodes $[i, j]$, $[k, l]$ can be linked if, and only if, $i \leq k$ and $j \leq l$, or, $i > k$, $j \leq l$ and $(i - k) \leq (l - j)$. Using this characterization of the set of potentially linkable pairs of nodes, a list P of cardinality 3 or 6, was generated randomly for the different instances.

The computational experiment was carried out on a personal computer with Intel ® Core (TM) i7-4720HQ, 2.60GHz with 16384 MB RAM. The algorithms were implemented and solved by using Xpress, Version 8.0.

2.4.1 Uncertain costs

The structure of the set of uncertain costs was designed following the idea of the uncertainty set of demand data proposed in Lee et al. (2013). The values f_{ij}^- and c_{ij}^- were generated randomly in the interval $[0, 1]$. For each arc $(i, j) \in A$, the fixed cost f_{ij} could take any value in the interval $[f_{ij}^-, f_{ij}^- + d_{ij}^f]$, and the unitary transportation arc cost c_{ij} belongs to the interval $[c_{ij}^-, c_{ij}^- + d_{ij}^c]$, where d_{ij}^f and d_{ij}^c denote the maximum possible deviations from the nominal cost values f_{ij}^- and c_{ij}^-, respectively. We use two parameters, Γ^f and Γ^c, to control the degree of uncertainty of fixed and unitary transportation arc costs, which must belong to the sets

$$F = \Big\{ f_{ij} = f_{ij}^- + d_{ij}^f v_{ij}^f, \quad \sum_{(i,j)\in A} v_{ij}^f \leq \Gamma^f,\ 0 \leq v_{ij}^f \leq 1,\ \forall (i, j) \in A \Big\},$$

and

$$C = \Big\{ c_{ij} = c_{ij}^- + d_{ij}^c v_{ij}^c, \quad \sum_{(i,j)\in A} v_{ij}^c \leq \Gamma^c,\ 0 \leq v_{ij}^c \leq 1,\ \forall (i,j) \in A \Big\}.$$

The values for the parameters Γ^f and Γ^c, controlling the uncertainty degree in costs, were obtained from a regular support of the interval $(0, 0.1)$ as a fraction of the number of the arcs having uncertain fixed cost or uncertain unitary transportation cost, specifically

$$\Gamma^f \in \{0.01, 0.03, 0.05, 0.07, 0.09\} \times n^f,$$

and

$$\Gamma^c \in \{0.01, 0.03, 0.05, 0.07, 0.09\} \times n^c,$$

where n^f and n^c represent the number of arcs with uncertainty in each type of cost. These values were chosen after observing that for smaller values of Γ^f and Γ^c the regret became almost 0, and for bigger values it remained almost constant. In fact, the values corresponding to the 0.09-fraction of n^f and n^c are not shown in the following tables 2.2 and 2.3 since they are very similar to the values obtained for the 0.07-fraction.

We generated different types of instances for the values of the parameters d_{ij}^f and d_{ij}^c. Table 5.1 shows the different discrete distributions used to generate instances at random. For example, for instances of type A, d_{ij}^f will take the value 0 with probability $\frac{3}{4}$, the value f_{ij}^- with probability $\frac{1}{8}$ or the value $2 \cdot f_{ij}^-$ with probability $\frac{1}{8}$ and d_{ij}^c will take the value 0 with probability $\frac{3}{4}$ or the value c_{ij}^- with probability $\frac{1}{4}$. Observe that we are allowing more variability in the fixed cost f_{ij} from their nominal cost value f_{ij}^- than in the unitary transportation cost c_{ij}, which can share certain similarities with practical applications in which the construction costs are more variable than the maintenance ones. Note also that in the successive types of instances the probability of an arc to have uncertain costs increases.

Instance type	d^f_{ij}	d^c_{ij}
A	$0 \cdot f^-_{ij}$, with probability $\frac{3}{4}$ $1 \cdot f^-_{ij}$, with probability $\frac{1}{8}$ $2 \cdot f^-_{ij}$, with probability $\frac{1}{8}$	$0 \cdot c^-_{ij}$, with probability $\frac{3}{4}$ $1 \cdot c^-_{ij}$, with probability $\frac{1}{4}$
B	$0 \cdot f^-_{ij}$, with probability $\frac{2}{3}$ $1 \cdot f^-_{ij}$, with probability $\frac{1}{6}$ $2 \cdot f^-_{ij}$, with probability $\frac{1}{6}$	$0 \cdot c^-_{ij}$, with probability $\frac{2}{3}$ $1 \cdot c^-_{ij}$, with probability $\frac{1}{3}$
C	$0 \cdot f^-_{ij}$, with probability $\frac{1}{3}$ $1 \cdot f^-_{ij}$, with probability $\frac{1}{3}$ $2 \cdot f^-_{ij}$, with probability $\frac{1}{3}$	$0 \cdot c^-_{ij}$, with probability $\frac{1}{3}$ $1 \cdot c^-_{ij}$, with probability $\frac{2}{3}$

Table 2.1: Types of instances.

We report in Tables 2.2 and 2.3 the type of instance (type), the number of nodes of each instance (n), the cardinality of the list of node pairs to be linked P ($\#P$) and the value of $\Gamma^f = \Gamma^c = \Gamma$ of the solved problems. For each configuration of instances and parameters we solved 5 different problems and reported the average values of the corresponding indicators; the optimal regret (R*), the CPU time (CPU), the number of iterations of the algorithm (It.), the percentage of arcs with uncertain cost c (among all the arcs with uncertain cost c) constructed in the optimal design ($\%c$) and the percentage of arcs with uncertain fixed cost f constructed in the optimal design ($\%f$).

We can observe that the average CPU time spent in solving each type of instance clearly increases with the increase of the number of nodes, the cardinality of P, and the value of Γ. For instance, in Table 2.3 we can observe that for type C, $n = 100$, $P = 3$, $\Gamma = 0.01$ the average CPU time is less than 6 seconds, meanwhile if we increase Γ to 0.07 the CPU time becomes 366 seconds (61 times higher).

Tables 2.2 and 2.3 also show the effect of the increase in the variability of the random distributions (Table 5.1), used to generate the range of costs, in the time needed to solve the problem. As might be expected the larger variability in the range

of costs, the larger CPU time needed to solve the problem.

It can also be observed in Tables 2.2 and 2.3 that the percentage of arcs having uncertain costs c_{ij} which are included in the robust design is bigger than the percentage of arcs with uncertainty in f_{ij}; this is caused by the type of considered instances. It can be seen in Table 5.1 that the range of variability allowed for costs f_{ij} is bigger than the one allowed for costs c_{ij}, and therefore including arcs with uncertain f_{ij} has a higher effect in the regret. For instance, for type B, $n = 196$, $|P| = 6$, $\Gamma = 0.03$ the percentage of arcs with uncertain c_{ij} included is 38.10%, meanwhile the percentage of arcs with uncertain f_{ij} included is 4.33%. This shows again the sensitiveness of the model to the structure of the uncertainty.

type	n	#P	Γ	R*	CPU	It.	%c	%f
A	100	3	0.01	0.35	0.36	2.20	47.37	6.08
A	100	3	0.03	89.82	0.81	4.00	47.37	5.78
A	100	3	0.05	110.78	1.23	5.00	47.68	5.78
A	100	3	0.07	116.00	1.00	5.00	47.69	6.06
A	100	6	0.01	0.19	0.62	2.20	51.29	7.91
A	100	6	0.03	47.99	1.68	4.20	50.72	6.65
A	100	6	0.05	54.52	2.00	4.20	50.72	6.65
A	100	6	0.07	56.00	1.00	4.00	50.00	6.35
A	196	3	0.01	2.74	1.56	2.80	51.55	1.68
A	196	3	0.03	65.95	2.79	4.40	51.86	0.92
A	196	3	0.05	66.21	2.39	4.00	51.39	1.38
A	196	3	0.07	66.00	2.00	4.00	51.16	1.53
A	196	6	0.01	93.82	4.79	5.20	47.64	3.55
A	196	6	0.03	187.68	7.97	8.20	47.96	3.09
A	196	6	0.05	189.82	7.47	7.80	47.96	3.09
A	196	6	0.07	190.00	8.00	8.00	48.03	3.10
A	225	3	0.01	41.14	2.71	4.00	49.59	2.30
A	225	3	0.03	135.33	5.53	7.00	49.59	2.17
A	225	3	0.05	139.61	6.39	7.60	49.59	2.17
A	225	3	0.07	140.00	6.00	8.00	49.66	1.91
A	225	6	0.01	66.82	6.53	5.00	48.63	3.76
A	225	6	0.03	166.72	9.73	6.40	48.63	3.63
A	225	6	0.05	173.84	13.95	7.80	48.50	3.49
A	225	6	0.07	174.00	11.00	7.00	48.37	3.36
B	100	3	0.01	74.73	2.51	7.00	41.18	4.72
B	100	3	0.03	205.79	6.93	12.40	41.39	3.07
B	100	3	0.05	215.08	4.89	10.80	41.39	2.83
B	100	3	0.07	215.00	6.00	11.00	41.30	2.35
B	100	6	0.01	56.13	0.96	3.80	38.29	6.44
B	100	6	0.03	217.71	2.95	8.00	37.78	5.56
B	100	6	0.05	269.03	2.81	8.00	38.29	5.56
B	100	6	0.07	273.00	3.00	8.00	37.97	5.56
B	196	3	0.01	71.99	3.36	4.80	36.13	1.53
B	196	3	0.03	162.90	4.62	6.00	36.02	1.41
B	196	3	0.05	165.09	4.81	5.60	36.02	1.41
B	196	3	0.07	165.00	4.00	5.00	36.21	1.18

Table 2.2: A-B instances solved by Algorithm 1.

type	n	#P	Γ	R*	CPU	It.	%c	%f
B	196	6	0.01	83.35	4.39	5.00	38.10	4.22
B	196	6	0.03	133.85	7.22	8.00	38.10	4.33
B	196	6	0.05	134.73	6.30	6.80	38.10	4.33
B	196	6	0.07	135.00	6.00	7.00	38.29	4.09
B	225	3	0.01	80.74	2.52	3.80	38.51	2.83
B	225	3	0.03	184.94	4.63	5.80	38.51	2.83
B	225	3	0.05	209.82	12.19	10.40	38.51	2.83
B	225	3	0.07	211.00	13.00	11.00	38.24	2.44
B	225	6	0.01	289.96	25.11	14.40	41.49	4.79
B	225	6	0.03	358.70	131.23	40.40	41.39	4.28
B	225	6	0.05	410.18	162.56	40.80	41.29	4.69
B	225	6	0.07	411.00	287.00	48.00	41.18	4.59
C	100	3	0.01	122.49	5.68	11.00	14.24	5.57
C	100	3	0.03	421.24	96.54	39.40	14.59	5.23
C	100	3	0.05	497.29	366.81	61.00	14.12	5.11
C	100	3	0.07	506.00	366.00	56.00	14.62	5.23
C	100	6	0.01	277.63	3.05	8.40	18.67	8.35
C	100	6	0.03	446.80	13.03	20.80	18.32	8.35
C	100	6	0.05	376.19	13.77	21.60	18.32	8.59
C	100	6	0.07	378.00	11.00	19.00	18.13	8.82
C	196	3	0.01	303.89	11.65	12.00	13.29	3.76
C	196	3	0.03	366.71	34.03	21.80	13.40	3.59
C	196	3	0.05	202.07	31.38	20.60	13.40	3.59
C	196	3	0.07	202.00	29.00	20.00	13.33	3.70
C	196	6	0.01	591.27	63.55	28.80	14.10	5.55
C	196	6	0.03	153.55	631.47	65.60	14.44	5.38
C	196	6	0.05	118.20	957.16	72.60	14.44	5.44
C	196	6	0.07	138.00	952.00	71.00	14.89	5.49
C	225	3	0.01	471.41	155.13	40.00	12.96	2.64
C	225	3	0.03	419.10	475.85	57.00	12.72	2.59
C	225	3	0.05	419.22	361.13	51.60	12.72	2.59
C	225	3	0.07	419.00	396.00	53.00	12.62	2.64
C	225	6	0.01	460.06	169.47	32.20	15.32	5.32
C	225	6	0.03	363.59	410.52	37.00	15.22	5.42
C	225	6	0.05	499.10	413.69	35.60	14.94	5.32
C	225	6	0.07	500.18	397.38	36.20	15.27	5.32

Table 2.3: B-C instances solved by Algorithm 1.

Finally, Tables 2.2 and 2.3 show clearly the positive correlation between the parameter Γ and the value of the optimal regret. For any given configuration of the instances, the increase in this parameter supposes a higher degree of uncertainty in the model, as a consequence of the definition of the polyhedra F and C, giving rise to the increase of the regret of any feasible design. For instance, for the configuration of type A, $n = 100$, $|P| = 3$, $\Gamma = 0.01$ the regret is 0.35, meanwhile if we increase Γ to 0.05 the average regret is 110.78.

2.4.2 Uncertain costs and linked pairs

The optimization model given by the formulation (RND) was enriched in Section 2.3.2 by enabling the possibility of considering uncertainty about the subset pairs of nodes of the list P that should be linked under a given scenario. Hence, for this model, one must compute the overall cost of constructing a network being able to connect any pair of nodes that potentially could have to be linked, while the transportation cost only has an effect in the objective function for those pairs of nodes for which it is mandatory to make its linkage under the considered scenario.

We show in Tables 2.4, 2.5 and 2.6, as in the precedent tables, the type of instance (A,B or C), its parameter configuration and the average of the CPU times, number of iterations and percentages of arcs with uncertain costs appearing in the optimal design. Here again, "% c" is the percentage corresponding to the arcs with uncertain unitary transportation cost included in the optimal design and "% f" the one corresponding to the arcs with uncertain fixed cost. Furthermore, we show in the column (g), the maximum number of pairs in P that must be linked under a given scenario. The parameter g is the right-hand side of the constraints (2.3) when $r = 1$ and in our experiment takes the values 1 and 2 if $|P| = 3$ and $1, 2, 3$ if $|P| = 6$.

As the degree of uncertainty increases in this model, so does the size of the corresponding formulation. Consequently, the CPU times spent in solving the new instances increase as it can be clearly observed comparing them with the corresponding entries of Tables 2.2 and 2.3. In fact, certain instances can not be solved within a given time limit. To highlight this point, we have added for each parameter configuration, two new values with the number of problems, out of five, solved to optimality within a limit of 1800 seconds (column "#opt"), and its average gap (column "Gap"). For example, in the *easiest* case, for instances of type A, $n = 36$, $P = 3$, $\Gamma = 0.01$, $g = 1$, we could not solve to optimality one of the problems, whereas for instances of type C, the most difficult ones, we could only solve to optimality 21 out of the 200 generated problems. The size of the considered instances stresses this effect, as can be seen comparing the instances with 36 and 49 nodes over the same table. When compared these new two columns among the three tables, it becomes evident the increase in the hardness of the problem as the result of the increase in the variability

of the generated uncertain costs, according to the probability distributions specified in Table 5.1.

Finally, we observed the behaviour that one may expect in what refers to the increase of the CPU times when the parameter g rises. The increase of such a value results in a bigger set of uncertain scenarios for the problem, making it more difficult to be solved. For example, for instances type A, $n = 49$, $P = 6$, $\Gamma = 0.07$, $g = 1, 2$, three problems out of five were solved to optimality, with an average gap of 29.60% and for $g = 3$ only one of the problems was solved and the average gap raised to 55.63%.

type	n	#P	Γ	g	#opt	CPU	Gap	It.	%c	%f
A	36	3	0.01	1	4	455.47	20.00	311.40	38.86	4.29
A	36	3	0.01	2	4	455.56	11.62	312.20	38.86	5.16
A	36	3	0.03	1	4	455.66	20.00	313.40	38.86	4.29
A	36	3	0.03	2	4	455.70	13.12	312.60	38.86	5.16
A	36	3	0.05	1	4	455.64	20.00	312.60	38.86	4.29
A	36	3	0.05	2	4	455.58	13.50	312.60	38.86	5.16
A	36	3	0.07	1	4	455.50	20.00	313.80	38.86	4.29
A	36	3	0.07	2	4	455.61	7.47	312.80	38.86	6.02
A	36	6	0.01	1	1	1800.00	64.03	712.00	48.67	14.42
A	36	6	0.01	2	2	1458.53	57.27	530.20	48.83	12.15
A	36	6	0.01	3	2	1458.80	60.00	532.20	50.89	11.15
A	36	6	0.03	1	1	1800.00	75.50	717.40	48.67	12.42
A	36	6	0.03	2	1	1816.99	74.77	712.80	49.78	14.25
A	36	6	0.03	3	1	1800.00	64.25	714.80	48.67	13.42
A	36	6	0.05	1	1	1800.00	63.01	719.40	49.78	10.31
A	36	6	0.05	2	1	1800.00	77.30	714.40	51.84	12.31
A	36	6	0.05	3	1	1800.00	72.96	716.80	51.84	11.31
A	36	6	0.07	1	1	1800.00	62.27	719.80	50.73	12.31
A	36	6	0.07	2	1	1800.00	66.49	720.60	50.89	12.31
A	36	6	0.07	3	2	1458.26	52.15	540.80	49.94	11.31
A	49	3	0.01	1	2	1176.00	52.91	848.40	52.05	4.93
A	49	3	0.01	2	2	1175.68	39.46	845.80	50.92	3.73
A	49	3	0.03	1	1	1535.86	68.98	1133.20	51.45	3.93
A	49	3	0.03	2	1	1536.27	59.18	1119.20	52.05	3.13
A	49	3	0.05	1	1	1535.76	70.32	1135.00	52.05	4.73
A	49	3	0.05	2	1	1535.88	58.59	1113.20	52.05	3.93
A	49	3	0.07	1	1	1535.81	68.00	1136.80	51.45	3.93
A	49	3	0.07	2	1	1535.75	60.71	1125.00	52.05	3.93
A	49	6	0.01	1	3	1102.89	33.57	353.00	50.98	11.98
A	49	6	0.01	2	3	1100.08	39.42	349.00	50.98	12.53
A	49	6	0.01	3	2	1458.78	48.56	523.80	50.98	13.16
A	49	6	0.03	1	2	1460.40	31.39	527.40	50.27	11.98
A	49	6	0.03	2	2	1459.32	31.04	526.20	51.38	13.49
A	49	6	0.03	3	1	1800.00	52.99	699.60	50.98	12.53
A	49	6	0.05	1	3	1102.66	29.86	350.60	50.27	11.98
A	49	6	0.05	2	2	1463.51	49.52	527.80	51.38	11.03
A	49	6	0.05	3	1	1800.00	59.20	703.40	52.29	11.65
A	49	6	0.07	1	3	1102.31	25.00	351.40	50.27	11.98
A	49	6	0.07	2	3	1101.47	29.60	348.40	51.38	10.47
A	49	6	0.07	3	1	1800.00	55.63	700.20	51.38	11.03

Table 2.4: Numerical results for Algorithm 2 with instances Type A.

type	n	#P	Γ	g	#opt	CPU	Gap	It.	%c	%f
B	36	3	0.01	1	4	456.02	20.00	312.80	40.42	6.50
B	36	3	0.01	2	4	456.46	20.00	314.00	40.42	6.50
B	36	3	0.03	1	3	816.30	39.50	621.40	40.42	6.50
B	36	3	0.03	2	3	816.44	39.99	618.40	39.81	6.50
B	36	3	0.05	1	3	816.10	39.99	625.40	40.42	7.27
B	36	3	0.05	2	3	816.25	39.62	625.00	39.81	5.73
B	36	3	0.07	1	2	1175.47	41.90	930.80	39.81	6.47
B	36	3	0.07	2	2	1175.90	41.26	932.00	39.81	6.50
B	36	6	0.01	1	1	1800.00	79.07	728.00	46.88	12.05
B	36	6	0.01	2	1	1800.00	64.98	724.00	47.06	11.43
B	36	6	0.01	3	1	1800.00	66.04	725.00	49.05	14.90
B	36	6	0.03	1	2	1460.05	57.81	543.00	48.43	13.03
B	36	6	0.03	2	1	1800.00	58.68	724.00	47.63	16.05
B	36	6	0.03	3	1	1800.00	67.81	727.20	49.11	13.61
B	36	6	0.05	1	2	1460.59	50.41	544.40	47.63	14.45
B	36	6	0.05	2	1	1800.00	52.44	719.60	47.06	10.76
B	36	6	0.05	3	1	1800.00	57.01	726.20	48.31	14.23
B	36	6	0.07	1	1	1800.00	69.47	719.60	47.00	12.85
B	36	6	0.07	2	0	1800.00	62.43	894.40	49.78	15.02
B	36	6	0.07	3	0	1800.00	61.31	893.20	48.98	15.64
B	49	3	0.01	1	3	816.11	29.03	559.20	41.76	8.46
B	49	3	0.01	2	3	815.97	29.31	565.80	41.76	7.55
B	49	3	0.03	1	2	1176.10	51.29	813.60	41.76	8.46
B	49	3	0.03	2	2	1175.99	43.96	836.20	42.28	7.94
B	49	3	0.05	1	2	1176.33	47.19	812.60	41.76	6.76
B	49	3	0.05	2	2	1175.96	51.91	829.80	41.76	7.28
B	49	3	0.07	1	2	1176.34	55.99	815.60	42.77	6.76
B	49	3	0.07	2	2	1176.15	50.09	837.20	42.28	7.28
B	49	6	0.01	1	2	1458.73	59.84	511.80	41.07	9.54
B	49	6	0.01	2	2	1458.22	58.84	505.00	40.44	9.12
B	49	6	0.01	3	2	1458.71	55.82	511.60	42.23	10.35
B	49	6	0.03	1	1	1800.00	50.48	681.40	43.42	7.91
B	49	6	0.03	2	1	1800.00	57.33	675.80	42.87	10.59
B	49	6	0.03	3	1	1800.00	66.32	677.40	42.23	10.34
B	49	6	0.05	1	1	1800.00	63.41	663.00	41.08	8.56
B	49	6	0.05	2	1	1800.00	56.90	672.20	43.40	8.31
B	49	6	0.05	3	1	1800.00	66.83	672.20	41.64	8.72
B	49	6	0.07	1	1	1800.00	52.44	674.60	41.67	9.54
B	49	6	0.07	2	1	1800.00	55.80	665.40	40.48	7.27
B	49	6	0.07	3	1	1800.00	67.07	669.40	42.80	7.90

Table 2.5: Numerical results for Algorithm 2 with instances Type B.

type	n	#P	Γ	g	#opt	CPU	Gap	It.	%c	%f
C	36	3	0.01	1	1	1535.06	76.76	1230.80	16.85	8.48
C	36	3	0.01	2	2	1175.61	52.66	909.40	16.51	10.07
C	36	3	0.03	1	1	1535.30	59.27	1218.40	16.81	8.90
C	36	3	0.03	2	1	1535.25	71.75	1224.80	15.79	9.69
C	36	3	0.05	1	1	1535.25	60.70	1206.40	16.80	9.03
C	36	3	0.05	2	1	1535.81	69.44	1237.20	17.87	9.64
C	36	3	0.07	1	1	1535.37	60.45	1214.60	16.80	9.03
C	36	3	0.07	2	1	1535.45	70.35	1226.00	16.50	9.09
C	36	6	0.01	1	0	1800.00	95.61	876.20	27.45	17.57
C	36	6	0.01	2	0	1800.00	88.23	858.40	29.59	20.09
C	36	6	0.01	3	0	1800.00	96.53	874.60	29.59	19.12
C	36	6	0.03	1	0	1800.00	70.56	859.00	28.44	19.08
C	36	6	0.03	2	0	1800.00	92.44	858.00	29.21	18.58
C	36	6	0.03	3	0	1800.00	91.76	865.00	28.87	18.73
C	36	6	0.05	1	0	1800.00	71.40	837.40	27.45	16.53
C	36	6	0.05	2	0	1800.00	79.61	838.20	27.79	16.67
C	36	6	0.05	3	0	1800.00	94.88	853.00	29.59	20.09
C	36	6	0.07	1	0	1800.00	61.14	834.60	29.27	19.43
C	36	6	0.07	2	0	1800.00	85.81	836.20	29.56	18.32
C	36	6	0.07	3	0	1800.00	80.82	840.40	28.13	17.39
C	49	3	0.01	1	2	1176.02	43.33	832.40	18.27	7.92
C	49	3	0.01	2	1	1535.62	74.10	1106.80	16.53	7.97
C	49	3	0.03	1	2	1176.59	42.72	833.40	18.02	7.92
C	49	3	0.03	2	1	1536.03	67.79	1105.20	16.80	9.15
C	49	3	0.05	1	1	1536.27	69.81	1071.20	16.56	7.48
C	49	3	0.05	2	2	1176.88	48.50	830.80	19.01	8.92
C	49	3	0.07	1	1	1535.30	75.65	1084.20	17.03	8.45
C	49	3	0.07	2	2	1176.03	55.42	829.20	17.02	8.21
C	49	6	0.01	1	0	1800.00	94.58	827.40	20.76	13.27
C	49	6	0.01	2	0	1800.00	99.49	822.20	19.69	13.74
C	49	6	0.01	3	0	1800.00	95.32	826.20	19.76	12.78
C	49	6	0.03	1	0	1800.00	70.65	784.60	21.56	13.25
C	49	6	0.03	2	0	1800.00	94.01	814.20	19.74	12.07
C	49	6	0.03	3	0	1800.00	94.56	825.00	19.53	13.74
C	49	6	0.05	1	0	1800.00	71.38	770.60	20.26	11.79
C	49	6	0.05	2	0	1800.00	88.23	798.00	23.78	14.20
C	49	6	0.05	3	0	1800.00	90.82	801.40	18.70	12.78
C	49	6	0.07	1	0	1800.00	72.66	756.60	19.96	11.57
C	49	6	0.07	2	0	1800.00	82.86	783.80	20.64	13.51
C	49	6	0.07	3	0	1800.00	92.62	804.80	21.62	14.47

Table 2.6: Numerical results for Algorithm 2 with instances Type C.

2.4.3 Uncertain costs, linked pairs and demands

To conclude this numerical experiment, we consider our most general model, developed in Section 2.3.3, in which apart from the costs and the pairs to be linked, there exists uncertainty about the demands to be served through each linkage.

As observed in the last set of generated instances in our experiment, the hardness of the problem prevents our algorithm from solving a large number of instances within a reasonable CPU time limit (specially for the Type C). If this was the observed behaviour under the assumption of uncertain costs and pairs to be linked, one may expect this situation does not become better when the uncertainty in served demands is also incorporated into the model. The difficulty of solving experimentally this problem was already observed in the literature for other similar design optimization models. For example, Lee et al. (2013), utilized in their computational study two test instances from real-life telecommunication network design problems, with 24 and 27 nodes respectively, that by that moment remained unsolved even for the deterministic case.

The model proposed in Section 2.3.3 is very general, having many different unknown parameters modeling the existing uncertainty about costs, pairs to be linked and flows. Furthermore, it combines uncertainty in a polyhedron of costs with discrete uncertainty for the pairs to be linked and demands to be served. This fact makes the model harder to be solved, even for moderate sizes, using conventional capabilities of computation. For this reason, we have decided to explore in detail just a numerical instance of the model instead of showing a table with CPU times and other indicators for a few small instances. This numerical example represents a real optical network topology with simulated parameters used in the literature (see for example Altın et al. (2007); Buchheim et al. (2011); Klinkowski et al. (2005)). The chosen network is the NSFNET, a USA backbone network with 14 nodes and 21 bi-directional links, depicted in Figure 2.5.

For this instance we generated at random the uncertain intervals for all the considered arc costs c_{ij} and f_{ij}. We defined 26 different pairs of nodes that potentially may be linked (bi-directionally): WA and DC with all the remaining nodes in the network. The possible demands to be sent across any potential linkage were assumed to be $0, 1$ or 2. Remind that 0 means that no demand is served, 1 can be understood as the usual demand and 2, as twice the usual demand. Furthermore, the connections (WA,DC), (WA,UT) and (GA,DC) were considered as potential arcs to be constructed (depicted as dashed arcs in Figure 2.6). Finally, we assumed that the exact number of pairs to be linked was not known, but it was known that at most five of them could be linked.

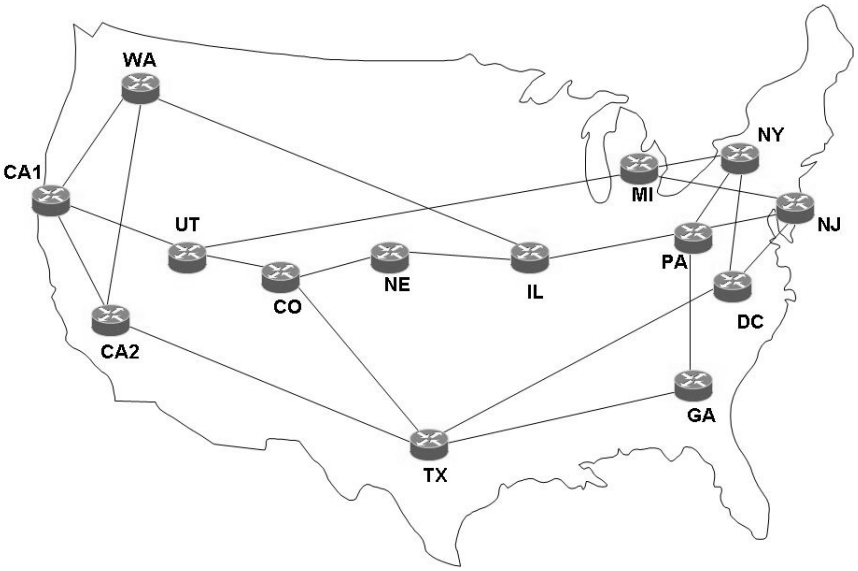

Figure 2.5: The NSFNET network topology. [1]

The regret obtained for this instance was 13.92. The CPU time in seconds required for solving the instance was 193094 (approximately 54 hours!) with 489 iterations of our proposed algorithm. We show in Figure 2.7 an optimal network design for this problem, where only 13 (highlighted with thicker lines) of the 21 bidirectional arcs were included in the optimal network design. We can also observe that the three potential arcs (WA,DC), (WA,UT) and (GA,DC) were constructed, extending in this way the existing optical network.

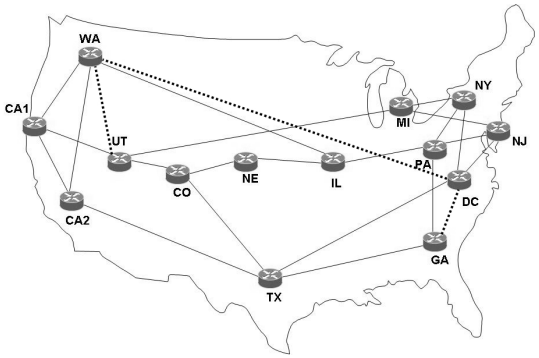

Figure 2.6: New potential arcs in the NSFNET network topology.

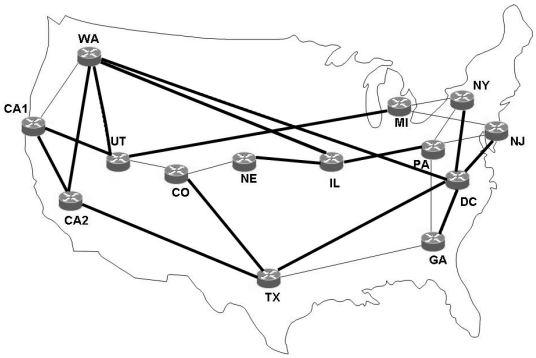

Figure 2.7: An optimal design for the NSFNET network topology.

After analysing the results obtained through the numerical experiment described in this section we may conclude that our methodology can be used in the design of

[1] Image from *https* : *//www.dit.upm.es/vnumlwiki/index.php/Example* − *NSF* − 14_1.8

robust distribution networks in the presence of a high level of uncertainty. Furthermore, using appropriate computational resources one can solve realistic approaches using different ways of modeling the existing uncertainty. By comparing the corresponding optimal solutions to different uncertainty models one may provide to the decision makers with a useful tool to choose robust designs of expensive infrastructures.

2.5 Conclusions

The models studied in this chapter help to built robust supply networks connecting pairs of (*distant*) nodes with a competitive performance under the uncertainties affecting the considered activity. The mathematical modeling of the problem using a scenario-based representation of the uncertainty can also help to design distribution networks when there is no historical database that can be used to formulate the problem through an approximation to the statistical distribution of the involved costs. This is the case, for instance, of humanitarian logistic networks that must assist the population affected by a specific natural disaster. These techniques can also be appropriate when we need a solution which will be implemented in unstable and changeable environments as commented in the context of the *agile manufacturing*. In these situations, the volatile nature of the conditions faced by the distribution network, weakens confidence in the use of stochastic modeling which need some stability to ensure the approximation of the average costs to the expectation of the statistical distributions in use for these costs (Strong Law of Large Numbers).

The minmax regret optimization models considered in this chapter have been solved by using a Benders decomposition framework. This methodology has shown to be flexible enough to be adapted to the different sources of uncertainty. The computational experiment carried out in the chapter, seem to support the applicability of the approach to real problems, although we must be realistic concerning to the certification of optimality. The increase in size of the considered networks will inevitably drive us to fail in the certification of the optimality using our numerical algorithm. However, even prematurely stopped, this numerical procedure still has the ability of providing an approximation to the optimal design as well as an upper bound on the incurred gap. The quality of such an approximation can be improved if the initial design used in the algorithm were obtained by solving the design optimization problem under one or several scenarios of costs/demands that could ensure a given degree of *proximity* to the optimal value (see Section 1.1.1). This could be a future research line aiming to guarantee the initialization of the numerical scheme with a *good* solution. Other further research lines could be the integration of these robust design models in other more general supply chain management problems. In

this context, new uncertainty sources could be considered in the production process. Moreover, cost structures of a more general nature could be used to model, for instance, the effect of the traveled distances in the quality of the transported commodities or dependencies of the uncertain demand of a given node respect to the supplier assigned.

Chapter 3

A minmax regret model for a time-dependent shortest path problem under uncertainties

We consider a shortest path problem where the arc costs depend on their relative position on a given path and there exist uncertain cost parameters. We study a min-max regret version of the problem under different types of uncertainty of the involved parameters. First, we provide a Mixed Integer Linear Programming formulation by using strong duality in the uncertainty interval case. Second, we develop three algorithms based on Benders decomposition for general polyhedral sets of uncertainty. In order to speed up the algorithms we use constant factor approximations to initialize them. Finally, we report some computational experiments for different uncertainty sets in order to compare the behavior of the proposed algorithms.

3.1 Introduction

Certain real applications require deciding the *best* schedule plans in which the cost of processing a task depends on its starting time. In many cases, we are given a set of planning tasks, usually represented as the arcs of a network that may model precedence relationships between them, while the goal is to determine the instant in which each task (arc) should start its processing (is traversed) in order to optimize some objective function. Natural examples are timetabling and scheduling problems (Borndörfer and Schlechte (2007); Cacchiani et al. (2010); Cacchiani and Toth (2012); Fischer and Helmberg (2014); Gawiejnowicz (2008)), Shortest Path Problems (SPP) in dynamic networks (Ahuja et al. (2003); Aronson (1989); Cooke and Halsey (1966); Fischer and Helmberg (2014); Hashemi et al. (2010)) or Time-Dependent Traveling Salesman Problems (TDTSP) (Bigras et al. (2008); Furini et al. (2016); Pessoa et al. (2010); Picard and Queyranne (1978); Taş et al. (2016); Vander Wiel and Sahinidis (1996)), among others.

An interesting type of time-dependency that we would like to emphasize here and that appears in some of the above TDTSP models (Bigras et al. (2008); Picard and Queyranne (1978); Vander Wiel and Sahinidis (1996)) is that of defining the travel cost between the two nodes of any arc as a function of the order in which the traveler moves from one of them to the other into the route. More specifically, the travel cost from node i to node j depends on the position that the node i occupies in the ordering sequence of visited nodes. This problem with discrete travel times can be viewed as a single machine scheduling problem with sequence-dependent setup times (Bigras et al. (2008); Picard and Queyranne (1978)).

The time-dependent SPP (TDSPP) has been considered under different optimality criteria and although there are models that can be solved in polynomial time (Ahuja et al. (2003)), their complexity level is usually very high. The size of an associated network called sometimes *time-expanded network* (Fischer and Helmberg (2014)), *multipartite network* (Bigras et al. (2008)) or *layered graph* (Abeledo et al.

(2013)) plays an important role in this level of complexity. This network is the basis of one of the most useful modeling approaches for discrete planning. Depending on the number of arcs and discretized time periods, the full time-expanded network easily grows to a huge number of arcs in actual applications. In order to handle this behavior Fischer and Helmberg (2014) developed a general dynamic graph generation framework to control the size of the network, what is crucial in order to obtain efficient algorithms to solve exactly the corresponding problem.

In this chapter, we use a time-expanded network as the basis to obtain a Linear Programming (LP) formulation of time-dependent SPP models where the cost structure takes into account the position that each arc occupies in the path, specifically, the total cost associated to each arc is the sum of an amount proportional to the position of the arc in the path and another quantity associated to the arc independently of its position. Furthermore, it is assumed that there exists uncertainty in these two cost coefficients and we propose a robust -minmax regret- optimization model in order to obtain a *good* solution. The complexity of the resulting optimization problem will be related to the region modeling the uncertainty and to the type of dependency on time of the cost structure. In most cases, this complexity will be NP-hard, for instance when the uncertain parameters are modeled through uncertainty intervals the TDSPP includes in particular the standard minmax regret SPP, which is known to be NP-hard (see Section 1.1.1).

Although time-dependent and robust SPP models have been studied separately in the literature (see e.g. Ahuja et al. (2003) and Karasan et al. (2001); Montemanni and Gambardella (2004), respectively), to the best of our knowledge this is the first time that a model, combining the time-dependency and the uncertainty of the costs, is addressed in the context of the minmax regret criterion. As we will see, combining both aspects will have as a consequence the emergence of relationships between the uncertain parameters which breaks the independent variation assumption usually considered in this context. In this chapter we will show how to find a MILP formulation for the problem assuming uncertainty intervals for the unknown parameters. Generalizing the structure of these uncertainty sets will require new developments in order to extend previous results and to accommodate the new conditions on the sources of uncertainty. We will use a Benders decomposition algorithm to solve numerically the resulting minmax regret TDSPP problem. Moreover, the proposed algorithm can easily be adapted to more general contexts as, for instance, that of the minmax regret TDTSP.

An interesting application of this last optimization model can be found in the context of scheduling. As said earlier, some authors (Bigras et al. (2008); Picard and Queyranne (1978)) have applied the TDTSP to model the problem of finding the optimal ordering in which a set of jobs must be processed in a single machine

to minimize the total completion time. In actual scheduling applications, at times, the parameters describing the problem are only partially known. This uncertainty justifies finding robust solutions. For instance, in the simple case in which no precedence order between the tasks is considered and the processing times are uncertain, the shortest processing time (SPT schedule), optimal for the total completion time criterion, may vary from scenario to scenario. This is a serious drawback for decision makers since their actions will possibly be evaluated ex-post and it would be really worthy to have an ex-ante robust schedule as the one proposed by the minmax regret criterion.

The minmax regret TDSPP model can also have interesting applications in the field of scheduling under uncertainty. In TDTSP applications one must find an optimal circuit in a network representing all the feasible processing orderings. Analogously, in the TDSPP model the goal is to determine an ordered sequence of a subset of tasks. In a context of uncertain processing times, a minmax regret TDSPP optimal solution will ensure a *competitive* total completion time under any of the considered scenarios. This type of solutions can be used as the basis of a procedure that sequentially assigns subsets of tasks to a set of available machines. In such a procedure, once the first sequence of tasks (a path) has been assigned to one of the machines, they (the tasks) would be removed from further consideration. Then, a new network would be built and the process repeated with the remaining tasks not yet processed. Similar schemes are used in *Grid Computing* (see, for instance He et al. (2003)) to assign feasible sequences of tasks to a set of available processors. In a grid computing architecture, a set of processors share a set of tasks according to certain technical characteristics. Some of them, as the required processing time of the tasks or the transfer speed together with the required computing recourses, could be uncertain. Most of this problems studied in the literature have a high level of difficulty since they are modeled as multiobjective combinatorial optimization problems (see for instance Ma et al. (2011a,b) and the references therein). This inherent difficulty increases when the problem is considered in an uncertainty context, which makes the construction of heuristic assignment procedures, like the one suggested above, a valuable tool.

In the following Section 3.1.1, we give another application of the TDSPP, based on the design of a searching strategy to find a hidden object in a finite set of possible locations. We present this application only for illustrative purpose. This optimization model will be used along the chapter as a conductive example to write different formulations for its minmax regret version under uncertain costs, to show dependency relations between these cost parameters and to complete some of the numerical experiments reported at the end of the chapter. In Section 3.2 we give an LP formulation of the TDSPP using a time-expanded network. As an application,

a formulation of the searching problem is proposed. In Section 3.3.1, it is shown how the dual of this LP problem allows us to obtain a MILP formulation for the minmax regret TDSPP under uncertainty intervals of costs. We extend here previous results on the minmax regret SPP to the specific cost structure in which the two types of involved costs vary independently into their corresponding uncertainty intervals. A new and more general case in which we consider polyhedral sets of uncertainty for these costs is also studied in this section, where three different Benders decomposition algorithms are analyzed. Although Benders decomposition has been applied to SPP in the past (see e.g. Montemanni and Gambardella (2005b)), we have developed a specific Branch and Cut procedure with Benders cuts reinforced with a set of initial cuts which are obtained from constant factor approximations to the problem (Conde (2010); Kasperski and Zielinski (2006)). In the last section of this chapter, the computational results of a numerical experiment including different uncertainty models for the searching problem are shown. In this experiment we have also compared different strategies for the Benders scheme with the exact resolution of the problem using standard optimization software.

3.1.1 An illustrative Example

We will describe here a new application of the TDSPP in the form of a game in which a hidden item is searched in a finite set of possible locations. This item could be some material object or not, for example a piece of information searched within a set of databases where one has to decide which databases are processed and which are not. The sequence followed over the locations to search for the item is determined by the accessing costs and the rewards obtained in the case of finding the object. Obviously, practical situations may include additional aspects to this basic optimization model that could substantially increase its difficulty. However, even in those cases, the resulting model could still share some of the essential characteristics described here.

Suppose that we need to find an object that can be located in one of n possible positions with probability p_i, $i = 1, \ldots, n$. If the object is found at the position i one obtains a given reward r_i, otherwise the profit is zero. In the case that the object has been found at the position i, the searching process is stopped, otherwise we can continue searching at another position j or finish the process. In this model, the decision of searching at the position j, given that the object has not been found at the location i, can only be taken if the arc (i, j) exists. In addition, we assume that this decision implies that one must pay d_{ij} units to access to this location j from i. The *searching problem* consists in choosing a sequence of positions in which we will search this object according to the expected reward. This sequence is chosen a priori, that is, we do not use the information of the previously visited nodes in order to modify the searching strategy. Next, we formally describe the searching example.

Let us consider a connected digraph $G = (N, A)$ with a set of nodes $N = \{0, 1, \ldots, n+1\}$ and a set of arcs $A \subseteq N \times N$. Nodes 0 and $n+1$ represent, respectively, the starting point and the ending of the searching process. Any other i represents one of the n positions where the object may be searched. We assume that all the arcs $(0, i)$, $i = 1, \ldots, n+1$ and $(j, n+1)$, $j = 0, 1, \ldots, n$ belong to A, each with zero cost, $d_{0i} = 0$, $i = 1, \ldots, n+1$ and $d_{j(n+1)} = 0$, $j = 1, \ldots, n$. The inclusion of these arcs, at zero cost, enables us to start/end the searching process at any node. Any other arc in A has an associated cost d_{ij}. As it was described above, under these conditions the searching problem is equivalent to determining a shortest path in G between 0 and $n+1$ with the cost structure induced by the coefficients d_{ij} and r_i, for every possible i and j. For instance, if the resulting path is only composed of the arc $(0, n+1)$, then the optimal solution does not search for the object at all. This particular case can occur if the expected reward of finding the object does not compensate the expected cost of any searching process.

Let $\mathcal{P}$ denote the set of paths in G from 0 to $n+1$, and let us consider a path $(0, i_1), (i_1, i_2), \ldots, (i_{k-1}, i_k), (i_k, n+1)$. For notational convenience we take $i_0 = 0$, $i_{k+1} = n+1$ and $r_0 = r_{n+1} = 0$. Once a path has been chosen, the corresponding cost is a random variable taking the values

$$\sum_{s=0}^{t} d_{i_s i_{s+1}} - r_{i_{t+1}},\ t = 0, \ldots k-1$$

with 'a priori' probability $p_{i_{t+1}}$ of finding the object at the node i_{t+1}, where $p_0 = p_{n+1} = 0$, and the value

$$\sum_{s=0}^{k} d_{i_s i_{s+1}},$$

with a probability $1 - \sum_{t=1}^{k} p_{i_t}$ that the object is located in one of the non-visited nodes. After some algebra, the expected cost can be written as

$$\sum_{s=0}^{k} \left(d_{i_s i_{s+1}} - r_{i_{s+1}} p_{i_{s+1}} - d_{i_s i_{s+1}} \sum_{t=1}^{s} p_{i_t} \right). \tag{3.1}$$

Hence, the contribution of each arc (i_s, i_{s+1}) to the cost has two parts, an individual cost which is specific for the arc, $d_{i_s i_{s+1}} - r_{i_{s+1}} p_{i_{s+1}}$ minus the amount $d_{i_s i_{s+1}} \sum_{t=1}^{s} p_{i_t}$ that depends on the nodes visited previously to i_s in the path. In the case that $p_i = 1/n$, $i = 1...n$ (discrete uniform distribution) the latter contribution depends only on the position of the arc (i_s, i_{s+1}) into the path, that is, the expression (3.1) reduces to

$$\frac{1}{n} \sum_{s=0}^{k} \left(n d_{i_s i_{s+1}} - r_{i_{s+1}} - s d_{i_s i_{s+1}} \right). \tag{3.2}$$

Let us observe that each one of these addends can be written as

$$(n-s)d_{i_s i_{s+1}} - r_{i_{s+1}}. \tag{3.3}$$

Hence, in order to minimize the expression (3.2) a tentative searching path should locate its arcs with larger d_{ij}-costs in its last positions. In this way, these accessing costs would penalize the total cost as little as possible since they would be multiplied by the factor $(n-s)$ which is decreasing with respect to the position of the arc in the path. On the other hand, if the reward r_j is high enough, the node j, could be included in the searching path even if the cost of accessing this node is considerable, because the path will try to include all the arcs with negative contribution to the cost (3.3). Note at this point that, although the coefficients (3.3) could be negative and could give rise to negative cycles, an optimal solution of the problem can be found by means of a flow problem with unitary upper bounds on the arc flows. For further details on this point, the reader is referred to the discussion in Section 3.2.

Thus, taking into account that n is a constant, the proposed optimization problem would consist in determining a shortest path $x \in \mathcal{P}$, from 0 to $n+1$, with arc costs $c_{ij} + \alpha_{ij} posit_{ij}(x)$, where $c_{ij} = nd_{ij} - r_j$, $\alpha_{ij} = -d_{ij}$ and $posit_{ij}(x)$ is the position occupied by the arc (i,j) in the corresponding path $x \in \mathcal{P}$.

We will end this section setting out a very simple numerical instance of this searching problem which will be used to illustrate later formulations and results.

Example *Searching Game*

Consider the digraph $G = \{N, A\}$ with $N = \{1,2,3\}$ and $A = \{(1,2),(1,3),(2,3)\}$. We need to find an object that is located in one of the 3 nodes of G with probabilities $\frac{1}{3}$.

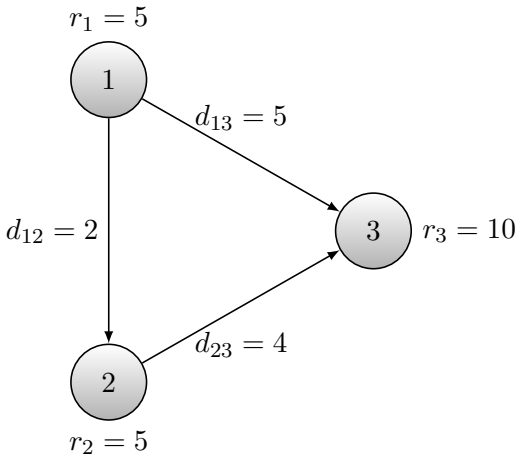

Figure 3.1: Searching network.

If the object is found at the positions 1 or 2 one obtains rewards $r_1 = r_2 = 5$, and if it is found at the position 3 a reward of $r_3 = 10$. The costs of crossing the three arcs $(1, 2), (1, 3)$ and $(2, 3)$ of the graph are $d_{12} = 2$, $d_{13} = 5$ and $d_{23} = 4$ respectively, as shown in Figure 3.1.

We add to the network of Figure 3.1, the node 0 representing the starting point of the searching process, the node 4 representing its ending, and the corresponding arcs $(0, i)$, $i = 1, 2, 3, 4$, $(j, 4)$, $j = 1, 2, 3$, all of them with cost 0 (Figure 3.2).

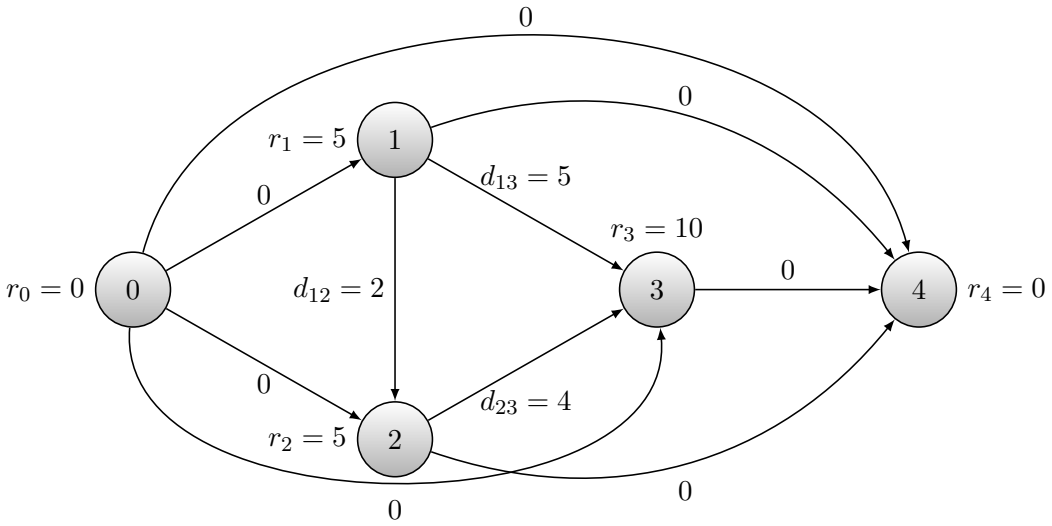

Figure 3.2: Searching problem network.

Then, from the definition of the searching problem, the choice of a sequence of positions in which we will search the object according to the expected reward is

equivalent to determining a shortest path from 0 to 4 with arc costs $c_{ij}+\alpha_{ij}posit_{ij}(x)$, where $c_{ij} = nd_{ij} - r_j$, $\alpha_{ij} = -d_{ij}$. Therefore, we have the following table of costs for the network of Figure 3.2

(i,j)	(0,1)	(0,2)	(0,3)	(0,4)	(1,2)	(1,3)	(1,4)	(2,3)	(2,4)	(3,4)
c_{ij}	-5	-5	-10	0	1	5	0	2	0	0
α_{ij}	0	0	0	0	-2	-5	0	-4	0	0

Table 3.1: Table of costs for the network of Figure 3.2.

□

3.2 Model description and deterministic case formulation

Let us consider a connected acyclic digraph $G = (N, A)$ with a set of nodes $N = \{0, 1, \ldots, n+1\}$ and a set of arcs $A \subseteq N \times N$. The acyclicity of G is instrumental in order to obtain a compact formulation of the problem. As mentioned above, if this condition were not fulfilled, a flow model with unitary upper bound on the flow traversing each arc could still be used in order to find an optimal path without cycles in $\mathcal{P}$. These upper bounds are compatible with the developments of Section 3.3.2 where some numerical methods are proposed but they are not with the structure of the time-expanded network used in Section 3.3.1 where a MILP formulation is obtained.

The problem here is to find a path in $\mathcal{P}$ with minimum cost according to a function that takes into account not only the costs c_{ij}, $(i,j) \in A$ but also a penalization proportional to α_{ij} associated to the position occupied for every arc in the ordering induced by the path. If x denotes a feasible path of $\mathcal{P}$ and (i,j) is one of its arcs we consider the following cost structure

$$c_{ij}(x) = c_{ij} + \alpha_{ij}posit_{ij}(x) \tag{3.4}$$

where c_{ij} is the fixed cost of traversing the arc (i,j) independently of its position in the path, $posit_{ij}(x)$ denotes the position of arc (i,j) in the path x, and α_{ij} is the unitary cost of traversing it in a given position on the path x.

It is possible to write a Mathematical Programming formulation for this problem by using a time-expanded network inspired on a similar one given in Picard and Queyranne (1978) for developing a Mixed Integer Linear Programming formulation for the TDTSP.

In what follows, we describe the time-expanded network $\overline{G}$ that fits to our optimization model. By $\overline{N}$ it is denoted the set of nodes, each one defined through

the pair (i,k) identifying that node i is the kth node to be visited following the ordering induced by a given path $x \in \mathcal{P}$, where $i \in N \setminus \{0, n+1\}$ and $k = 1, \ldots, n$. Two special nodes, $(0,0)$ and $(n+1, n+1)$, are added to the set $\overline{N}$ with a different meaning, the beginning and the end of every feasible path considered in $\mathcal{P}$.

The set of arcs, $\overline{A}$, of this new graph could be described by a vector of three indices (i,j,k), where $(i,j) \in A$ and $k \in \{0, 1, \ldots, n\}$, with the following meaning:

- If $i \neq 0$ and $j \neq n+1$ the arc described by the triplet (i,j,k), for all $k = 1, \ldots, n$, has the node $(i,k) \in \overline{N}$ as its origin and $(j, k+1) \in \overline{N}$ as its ending node.
- If $i = 0$ there are two possibilities:
  - $j \neq n+1$, in which case, the arc $(0,j,0)$ is defined in $\overline{A}$ with origin at $(0,0)$ and end at $(j,1)$.
  - $j = n+1$, in which case, the arc $(0, n+1, 0)$ is defined in $\overline{A}$ with origin at $(0,0)$ and end at $(n+1, n+1)$.
- If $j = n+1$, then $(n+1, n+1)$ is the ending node of the arc $(i, n+1, k)$ and (i,k) is its origin node for all $i, k \in N \setminus \{0, n+1\}$.

The cost associated to the three groups of the arcs $(i,j,k) \in \overline{A}$ is $c_{ij} + k\alpha_{ij}$, which corresponds to visiting node j right after visiting node i as the kth node in the ordering induced by the path. Note that, under this assumption, the cost of the arcs of the form $(0,j,0)$ is c_{0j}. In our instance of the searching problem described in Section 3.1.1, $c_{0j} = -r_j$, where r_j is the reward associated to the location j, since all the costs d_{0j} are zero.

The third group of arcs $(i, n+1, k)$ of the above description were not included in the multipartite networks considered in Bigras et al. (2008); Picard and Queyranne (1978). In our case, we need to include them in order to model the set of paths from the node 0 to $n+1$ in the original network G instead of a tour of a TSP as in Bigras et al. (2008); Picard and Queyranne (1978). Hence, when the arc $(i, n+1, k)$ is included in the path, the node i is visited in the kth place and after that the path ends at node $n+1$ of G.

Defining the flow variables x_{ij}^k taking value 1 when the arc $(i,j,k) \in \overline{A}$ is included in the path, and 0 otherwise, one has the following LP formulation for the Deterministic TDSPP

$$
\begin{array}{lll}
Z(c,\alpha) = & \min & Z(x,c,\alpha) := \displaystyle\sum_{(i,j,k)\in\overline{A}} x_{ij}^k (c_{ij} + k\alpha_{ij}) \\
& \text{st.} & \\
& & \displaystyle\sum_{(i,j,k)\in\overline{A}} x_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} x_{li}^{(k-1)} = b_i, \quad \forall (i,k) \in \overline{N}, \\
& & x_{ij}^k \geq 0, \qquad\qquad\qquad\qquad\quad \forall (i,j,k) \in \overline{A},
\end{array}
\tag{DTDSPP}
$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$.

Observe that, if the graph were not acyclic, the unitary upper bound constraints $\sum_{k:(i,j,k)\in\overline{A}} x_{ij}^k \leq 1, \forall (i,j) \in A$, could be included into the above formulation to ensure that there is no cycle in any feasible path. However, in this case, we must consider binary variables $x_{ij}^k \in \{0,1\}$ since the unimodularity property of the flow constraints is lost.

Example *Searching Game* (continuation)
We present next, the time-expanded network $\bar{G}$ that fits to the searching problem of our example.

We have the following set of nodes

$$\bar{N} = \{(0,0),(1,1),(2,1),(2,2),(3,1),(3,2),(3,3),(4,4)\}.$$

Observe that the node $1 \in N$ can only be visited in position 1, since we can only reach it from node 0; node $2 \in N$ can be visited in positions 1 and 2, from nodes 0 and $1 \in N$; and node $3 \in N$ in position 1, 2 and 3.

Figure 3.3 shows the time-expanded network with arc costs $c_{ijk} = c_{ij} + k\alpha_{ij}$.

We can easily observe that the shortest path is given by the path

$$(0,0),(1,1),(2,2),(3,3),(4,4)$$

with a total cost of -12. This sequence translated in terms of our searching problem means that:

1. the search process should start in node 1;
2. then, if the object is not found there, the search should be continued in node 2;
3. finally, if the object is not found in node 2, the search should end in node 3.

According to (DTDSPP), we can obtain the optimal path by solving the following LP formulation:

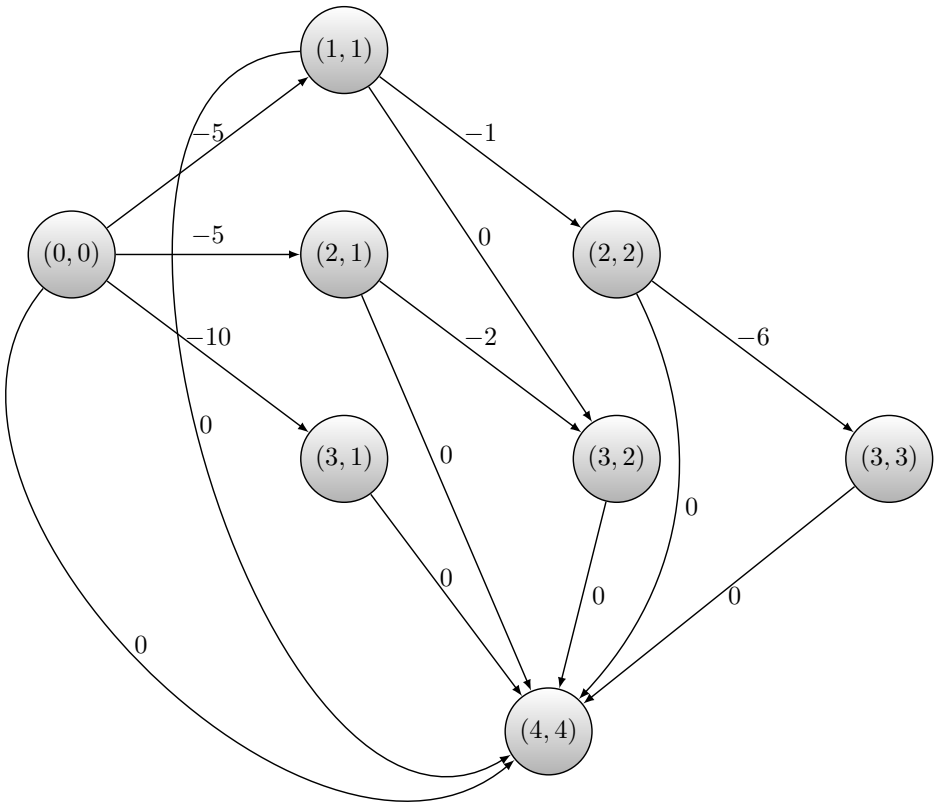

Figure 3.3: Time-expanded network.

$$
\begin{aligned}
Z(c,\alpha) = \quad & \min \quad Z(x,c,\alpha) := -5x_{01}^0 - 5x_{02}^0 - 10x_{03}^0 - 1x_{12}^1 - 2x_{23}^1 - 6x_{23}^2 \\
& \text{st.} \\
& \qquad x_{01}^0 + x_{02}^0 + x_{03}^0 + x_{04}^0 = 1, \\
& \qquad x_{01}^0 - x_{12}^1 - x_{13}^1 - x_{14}^1 = 0, \\
& \qquad x_{02}^0 - x_{23}^1 - x_{24}^1 = 0, \\
& \qquad x_{03}^0 - x_{34}^1 = 0, \\
& \qquad x_{12}^1 - x_{23}^2 - x_{24}^2 = 0, \\
& \qquad x_{13}^1 + x_{23}^1 - x_{34}^2 = 0, \\
& \qquad x_{23}^2 - x_{34}^3 = 0 \\
& \qquad -x_{04}^0 - x_{14}^1 - x_{24}^1 - x_{34}^1 - x_{24}^2 - x_{34}^2 - x_{34}^3 = -1, \\
& \qquad x_{ij}^k \geq 0, \quad \forall (i,j,k) \in \overline{A},
\end{aligned}
$$

□

3.3 Robust Model formulations and Benders algorithms

Once the model, i.e. the expanded network and the cost structure, is introduced, we assume that the coefficients of the objective function of Problem ($DTDSPP$) are imprecisely known and we propose a robust solution by using the minmax regret criterion. First, we will consider the model in which the uncertainty is modeled through a hypercube of coefficients c_{ij} and α_{ij}. In this case, we assume that *uncertainty intervals* $[c_{ij}^-, c_{ij}^+]$ and $[\alpha_{ij}^-, \alpha_{ij}^+]$ are known and each coefficient can vary independently of the rest. The set S given by the Cartesian product of these intervals defines the set of possible *scenarios* of costs. Then, we will consider a more general set of scenarios S in which we allow some dependencies between the uncertain parameters.

For a given path x in $\mathcal{P}$, from 0 to $n+1$, the regret under a cost scenario $(c,\alpha) \in S$ is defined as

$$R(x,c,\alpha) = Z(x,c,\alpha) - Z(c,\alpha), \tag{3.5}$$

where x identifies at the same time a path in $\mathcal{P}$ and the characteristic vector of the path that obviously satisfies the constraints of Problem ($DTDSPP$).

The problem that we consider is the one of finding a path in $\mathcal{P}$ minimizing the maximum regret over the whole set of scenarios S, that is, the Robust TDSPP:

$$R^* = \min_{x \in \mathcal{P}} \max_{(c,\alpha) \in S} R(x,c,\alpha). \tag{RTDSPP}$$

This problem will be called in what follows the minmax regret Time-Dependent

Shortest Path Problem. Observe that in the particular interval scenario case where all $\alpha_{ij}^+ = \alpha_{ij}^- = 0$ this model reduces to the standard minmax regret SPP, whose complexity is NP-hard even if the network is directed, acyclic, and has a layered structure (see Section 1.1.1), so that the same complexity is achieved in our model.

3.3.1 Hypercube uncertainty set

We assume in this first case that coefficients c_{ij} and α_{ij} can take any value in the intervals $[c_{ij}^-, c_{ij}^+]$ and $[\alpha_{ij}^-, \alpha_{ij}^+]$, respectively, independently of the rest. Then, set S is the hypercube determined by the Cartesian product of these intervals of costs.

As said before, one of the most common techniques used in the literature in order to solve the minmax regret SPP with uncertainty intervals of costs is based in determining a *worst-case* scenario (see Section 1.1.1). Remind that this is a scenario for which the maximum regret for a given path $x \in \mathcal{P}$ is reached, that is, $(c^w(x), \alpha^w(x)) \in S$ represents a worst-case scenario for the path x if it satisfies that

$$\max_{(c,\alpha)\in S} R(x, c, \alpha) = R(x, c^w(x), \alpha^w(x)).$$

For a given $x \in \mathcal{P}$ the set of scenarios verifying the above equation may have a cardinality greater than one but, at least, there exists one of these scenarios since, by definition, $R(x, c, \alpha)$ is convex as a function of $(c, \alpha) \in S$, hence it reaches its maximum at one of the extreme points of S. We denote one of these worst-case scenarios by $(c^w(x), \alpha^w(x))$.

Let us suppose that we can identify one of the worst-case scenarios for x by means of a linear function, namely, there exists a $2|A| \times |A|$-matrix B such that $(c^w(x), \alpha^w(x))' = Bx$. In this case, due to the unimodularity properties of the SPP, the value $Z(c^w(x), \alpha^w(x))$ can be obtained by solving the dual formulation of $(DTDSPP)$ in which the product Bx will appear in the right hand side of the constraints of the dual problem. This technique allows us to write Problem $(RTDSPP)$ using a MILP formulation as it has been done in the literature, for instance in Kasperski (2008) (Section 7.5) or in the paper Karasan et al. (2001). We can follow the same steps in our model if, for example, there is only uncertainty about the cost coefficients c_{ij}, that is, the parameters α_{ij} are considered crisp or nominal ($\alpha_{ij}^- = \alpha_{ij}^+$). In this case, by using Proposition 2, it is easy to obtain a worst-case scenario for a given $x \in \mathcal{P}$ as

$$c_{ij}^w(x) \quad = \quad c_{ij}^- + (c_{ij}^+ - c_{ij}^-) \left(\sum_{k:(i,j,k)\in\overline{A}} x_{ij}^k \right). \tag{3.6}$$

Since this expression is linear in the variables x we can use duality to obtain a MILP

formulation (Kasperski (2008); Karasan et al. (2001)).

However, for a given $x \in \mathcal{P}$, if the function

$$y \mapsto \max_{(c,\alpha)\in S} [Z(x,c,\alpha) - Z(y,c,\alpha)]$$

can be written as a linear function of y with its coefficients also depending linearly on x, it is still possible to apply LP duality in order to obtain a MILP formulation.

We will use this property to extend the classical duality development to obtain a MILP formulation when there exists uncertainty about the true values of both sets of costs, c_{ij} and α_{ij}, that is, the parameters α_{ij} are not longer crisp or nominal ($\alpha_{ij}^- \leq \alpha_{ij}^+$). Let us define the following set of linear functions of $x \in \mathcal{P}$,

$$\alpha_{ijk}^w(x) \quad = \quad k\alpha_{ij}^- + (\alpha_{ij}^+ - \alpha_{ij}^-) \left(\sum_{t:(i,j,t)\in\overline{A}} \min\{k,t\} x_{ij}^t \right). \tag{3.7}$$

Proposition 9. *The costs $c_{ij}^w(x)$ and $\alpha_{ijk}^w(x)$ for $(i,j,k) \in \overline{A}$, defined by* (3.6) *and* (3.7)*, verify*

$$\max_{(c,\alpha)\in S} [Z(x,c,\alpha) - Z(y,c,\alpha)] = \sum_{(i,j,t)\in\overline{A}} (c_{ij}^+ + t\alpha_{ij}^+) x_{ij}^t - \sum_{(i,j,k)\in\overline{A}} (c_{ij}^w(x) + \alpha_{ijk}^w(x)) y_{ij}^k$$

when the set of cost scenarios S is the hypercube defined by the uncertainty interval $[c_{ij}^-, c_{ij}^+]$ and $[\alpha_{ij}^-, \alpha_{ij}^+]$ for every $(i,j) \in A$.

Proof. By $(DTDSPP)$,

$$Z(x,c,\alpha) - Z(y,c,\alpha) = \sum_{(i,j,k)\in\overline{A}} c_{ij}(x_{ij}^k - y_{ij}^k) + \sum_{(i,j,k)\in\overline{A}} k\alpha_{ij}(x_{ij}^k - y_{ij}^k). \tag{3.8}$$

The first summation can be written as

$$\sum_{(i,j)\in A} c_{ij} \left[\left(\sum_{k:(i,j,k)\in\overline{A}} x_{ij}^k \right) - \left(\sum_{k:(i,j,k)\in\overline{A}} y_{ij}^k \right) \right]$$

therefore, as $x, y \in \mathcal{P}$, its maximum in $c_{ij} \in [c_{ij}^-, c_{ij}^+]$ is reached when $c_{ij} = c_{ij}^w(x)$ as defined in (3.6). Hence,

$$\max_{\substack{c_{ij} \in [c_{ij}^-, c_{ij}^+], \\ (i,j) \in A}} \sum_{(i,j,k)\in\overline{A}} c_{ij}(x_{ij}^k - y_{ij}^k) = \sum_{(i,j,t)\in\overline{A}} c_{ij}^+ x_{ij}^t - \sum_{(i,j,k)\in\overline{A}} c_{ij}^w(x) y_{ij}^k,$$

since, for any $x \in \mathcal{P}$, $\sum_{k:(i,j,k)\in\overline{A}} x_{ij}^k \in \{0,1\}$.

On the other hand, in order to maximize in α the second summation of (3.8) we

can observe that

$$\max_{\substack{\alpha_{ij} \in [\alpha_{ij}^-, \alpha_{ij}^+], \\ (i,j) \in A}} \sum_{k:(i,j,k)\in\overline{A}} k\alpha_{ij}(x_{ij}^k - y_{ij}^k) = \alpha_{ij}^+ \left(\sum_{t:(i,j,t)\in\overline{A}} t x_{ij}^t - \sum_{k:(i,j,k)\in\overline{A}} k y_{ij}^k \right) - \tag{3.9}$$

$$-(\alpha_{ij}^+ - \alpha_{ij}^-) \left(\sum_{k:(i,j,k)\in\overline{A}} \left(\sum_{t:(i,j,t)\in\overline{A}} \min\{k,t\} x_{ij}^t - k \right) y_{ij}^k \right), \tag{3.10}$$

where (3.10) is nonzero only if there exists $k > \sum_{t:(i,j,t)\in\overline{A}} t x_{ij}^t$ such that $y_{ij}^k = 1$. Using this last expression the result follows.

□ □

Based on the construction of the network $\overline{G}$ described in the previous section, one can write the dual formulation of the SPP problem $(DTDSPP)$ as

$$\begin{aligned} Z(c,\alpha) = \quad & \max \quad u_{00} - u_{(n+1)(n+1)} \\ & \text{st.} \\ & \qquad u_{ik} - u_{j(k+1)} \leq c_{ij} + k\alpha_{ij}, \quad \forall (i,j,k) \in \overline{A}, \end{aligned} \tag{3.11}$$

where the variable u_{00} can be fixed to 0 since the structure of this problem makes that any feasible solution can be identified with a ray of solutions obtained by adding the same constant to all its components.

On the other hand, the problem $(RTDSPP)$ can be written as

$$R^* = \min_{x\in\mathcal{P}} \max_{y\in\mathcal{P}} \max_{(c,\alpha)\in S} \left[Z(x,c,\alpha) - Z(y,c,\alpha) \right]. \tag{3.12}$$

Therefore, by using Proposition 9 and the expressions (3.6), (3.7), and using an approach similar to that in Karasan et al. (2001), a MILP formulation for problem $(RTDSPP)$ on acyclic networks can be written as follows

$$\begin{aligned} R^* = \quad & \min \quad \sum_{(i,j,t)\in\overline{A}} (c_{ij}^+ + t\alpha_{ij}^+) x_{ij}^t - u_{00} + u_{(n+1)(n+1)} \\ & \text{st.} \\ & \qquad \sum_{(i,j,k)\in\overline{A}} x_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} x_{li}^{(k-1)} = b_i, \qquad \forall (i,k) \in \overline{N}, \\ & \qquad u_{ik} - u_{j(k+1)} \leq c_{ij}^w(x) + \alpha_{ijk}^w(x), \qquad \forall (i,j,k) \in \overline{A}, \\ & \qquad x_{ij}^k \in \{0,1\}, \qquad \forall (i,j,k) \in \overline{A}, \end{aligned} \tag{3.13}$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$ and $c^w(x)$, $\alpha^w(x)$ are given in (3.6), (3.7).

The above MILP formulation does not apply in general to the searching problem described in the section 3.1.1. The reason is the dependency relationship between

the set of costs c_{ij} and α_{ij} when there exists uncertainty about the true values of the distance costs d_{ij}. However, the worst-case scenario (3.6) can be adapted to this model when it is only assumed uncertainty about the set of rewards r_j. In the latter situation we have the following specifications

- $c_{ij} = nd_{ij} - r_j$ and $\alpha_{ij} = -d_{ij}$,
- r_j is an uncertain parameter that can take on any value in the interval $[r_j^-, r_j^+]$ for every $j = 1, \cdots, n$,
- $c_{ij}^- = nd_{ij} - r_j^+$ and $c_{ij}^+ = nd_{ij} - r_j^-$ for all $(i,j) \in A$,
- $(0,j) \in A$ and $(j, n+1) \in A$ for all $j = 1, \cdots, n$,
- $c_{0j}^+ = -r_j^-$, $c_{0j}^- = -r_j^+$ and $c_{j,n+1}^{\pm} = \alpha_{j,n+1} = 0$ for all $j = 1, \cdots, n$,
- the remaining parameters are known (crisp or nominal).

By using Proposition 2, it is easy to obtain a worst-case scenario for $x \in \mathcal{P}$ given by the linear expression

$$c_{ij}^w(x) = nd_{ij} - r_j^+ + (r_j^+ - r_j^-) \left(\sum_{i,k:(i,j,k)\in\overline{A}} x_{ij}^k \right). \tag{3.14}$$

Finally, using (3.14) and taking $\alpha_{ijk}^w(x) = -d_{ij}$ for all k and $x \in \mathcal{P}$, we obtain with (3.13) a valid MILP formulation for this model.

Example *Searching Game* (continuation)
We continue now with our example assuming that each reward is uncertain, and can take any value in the following intervals $r_1 \in [3,6]$, $r_2 \in [4,6]$ and $r_3 \in [10,30]$. The corresponding uncertainty intervals $c_{ij} \in [nd_{ij} - r_j^+, nd_{ij} - r_j^-]$ are shown in Table 3.2.

(i,j)	(0,1)	(0,2)	(0,3)	(0,4)	(1,2)	(1,3)	(1,4)	(2,3)	(2,4)	(3,4)
$[c_{ij}^-, c_{ij}^+]$	[-6,-3]	[-6,-4]	[-30,-10]	0	[0,2]	[-15,5]	0	[-18,2]	0	0
α_{ij}	0	0	0	0	-2	-5	0	-4	0	0

Table 3.2: Uncertainty intervals for the c_{ij} costs and values for α_{ij}.

Therefore, following (3.14), we can write the next MILP formulation that models

the minmax regret time-dependent searching problem:

$$
\begin{aligned}
R^* = \min \quad & -3x_{01}^0 - 4x_{02}^0 - 10x_{03}^0 + 0x_{12}^1 + 0x_{13}^1 - 2x_{23}^1 - 6x_{23}^2 - u_{00} + u_{44} \\
& \text{st.} \\
& x_{01}^0 + x_{02}^0 + x_{03}^0 + x_{04}^0 = 1, \\
& x_{01}^0 - x_{12}^1 - x_{13}^1 - x_{14}^1 = 0, \\
& x_{02}^0 - x_{23}^1 - x_{24}^1 = 0, \\
& x_{03}^0 - x_{34}^1 = 0, \\
& x_{12}^1 - x_{23}^2 - x_{24}^2 = 0, \\
& x_{13}^1 + x_{23}^1 - x_{34}^2 = 0, \\
& x_{23}^2 - x_{34}^3 = 0 \\
& -x_{04}^0 - x_{14}^1 - x_{24}^1 - x_{34}^1 - x_{24}^2 - x_{34}^2 - x_{34}^3 = -1, \\
& u_{00} - u_{11} \leq -6 + 3x_{01}^0, \\
& u_{00} - u_{21} \leq -6 + 2x_{02}^0, \\
& u_{00} - u_{31} \leq -30 + 20x_{03}^0, \\
& u_{00} - u_{41} \leq 0, \\
& u_{11} - u_{22} \leq 2x_{12}^1 - 2, \\
& u_{11} - u_{32} \leq -15 + 20x_{13}^1 - 5, \\
& u_{21} - u_{32} \leq -18 + 20(x_{23}^1 + x_{23}^2) - 4, \\
& u_{22} - u_{33} \leq -18 + 20(x_{23}^1 + x_{23}^2) - 8, \\
& u_{11} - u_{42} \leq 0, \\
& u_{21} - u_{42} \leq 0, \\
& u_{31} - u_{42} \leq 0, \\
& u_{22} - u_{43} \leq 0, \\
& u_{32} - u_{43} \leq 0, \\
& u_{33} - u_{44} \leq 0, \\
& x_{ij}^k \geq 0, \quad \forall (i,j,k) \in \overline{A},
\end{aligned}
$$

□

As we have seen in this section, under a specific structure of the uncertain parameters of the model, we can obtain a MILP formulation and hence, we can solve exactly the problem for instances of *moderate* size. However, the independent variation of the uncertain parameters in the set of scenarios could be unrealistic in some applications. For instance, in the searching problem, as we have just seen, if the rewards r_j and the accessing costs d_{ij} are all uncertain it is not possible to apply Proposition 9 to get the above MILP formulation. In the following section we will consider an alternative way of solving a more general problem.

3.3.2 Polyhedral uncertainty set

In this section, we analyze the minmax regret TDSPP problem ($RTDSPP$) in a more general context. Now, we will permit that the uncertainty affects and links simultaneously both coefficients c_{ij} and α_{ij} of (3.4). This new framework allows us, for instance, to consider the searching problem of the section 3.1.1 under simultaneous variations of the rewards r_j and accessing costs d_{ij}, for every $(i,j) \in A$. Recall that, in spite of the fact that the following formulations will be given for acyclic graphs, they can be extended to not acyclic graphs by including the unitary upper bound constraints described in Section 3.2. Furthermore, the approach developed in this section can also be applied to the minmax regret TDTSP problem by adapting the constraints of the formulations described here to the structure of the TSP.

In order to model the set of possible scenarios, it is assumed that $(c,\alpha) \in S$, where $S \subset \mathbb{R}^{2|A|}$ is a bounded polyhedron.

In general, when an arbitrary polyhedron models the set of possible scenarios, one does not have a closed-form for a worst-case scenario that could be used to reformulate the problem. In this situation, we need to determine numerically a worst-case scenario for each given solution $x \in \mathcal{P}$. Solving the following formulation (3.15), one can find simultaneously a worst-case scenario for a given x and additionally an optimal solution to the underlying deterministic model for such a scenario, a *worst-case adversary*,

$$
\begin{aligned}
R(x) = \quad & \max \quad \sum_{(i,j,k)\in\overline{A}} x_{ij}^k (c_{ij} + k\alpha_{ij}) - \sum_{(i,j,k)\in\overline{A}} y_{ij}^k (c_{ij} + k\alpha_{ij}) \\
& \text{st.} \\
& \qquad \sum_{(i,j,k)\in\overline{A}} y_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} y_{li}^{(k-1)} = b_i, && \forall (i,k) \in \overline{N}, \\
& \qquad y_{ij}^k \in \{0,1\}, && \forall (i,j,k) \in \overline{A}, \\
& \qquad (c,\alpha) \in S,
\end{aligned}
\tag{3.15}
$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$.

Under our hypothesis all the constraints of (3.15) are linear functions of the decision variables (y,c,α). Nevertheless, we have a quadratic objective function and the binary variables y_{ij}^k. This last inconvenience comes from the fact that the constraints of (3.15) are, in general, not totally unimodular since we are considering general polyhedra S to model the set of possible cost scenarios.

In what follows the quadratic objective function of the problem (3.15) will be linearized using that S is a bounded set. This linearization makes possible to write the problem as a MILP formulation which opens the possibility to apply off-the-shelf solvers in order to have an optimal solution.

Let us define

$$M = \max_{(c,\alpha)\in S} \|(c,\alpha)\|_\infty. \tag{3.16}$$

Such a constant M exists since S is bounded. We add now the new decision variables z_{ij}^k, w_{ij}^k, $\forall i,j,k$, and the following constraints

$$\begin{aligned} -My_{ij}^k \leq \quad & z_{ij}^k \quad \leq c_{ij} + M(1-y_{ij}^k), \\ c_{ij} - M(1-y_{ij}^k) \leq \quad & z_{ij}^k \quad \leq My_{ij}^k, \\ -My_{ij}^k \leq \quad & w_{ij}^k \quad \leq \alpha_{ij} + M(1-y_{ij}^k), \\ \alpha_{ij} - M(1-y_{ij}^k) \leq \quad & w_{ij}^k \quad \leq My_{ij}^k, \end{aligned}$$

$\forall i,j,k$, to formulation (3.15).

On the one side, observe that if $y_{ij}^k = 0$ the above constraints imply that $z_{ij}^k = 0$ and $w_{ij}^k = 0$. On the other hand, if $y_{ij}^k = 1$ we obtain that $z_{ij}^k = c_{ij}$ and $w_{ij}^k = \alpha_{ij}$. Therefore, making $z_{ij}^k = c_{ij}y_{ij}^k$ and $w_{ij}^k = \alpha_{ij}y_{ij}^k$, $\forall i,j,k$ in formulation (3.15) we obtain an equivalent MILP formulation.

$$\begin{aligned} R(x) = \quad & \max \quad \sum_{(i,j,k)\in\overline{A}} x_{ij}^k(c_{ij} + k\alpha_{ij}) - \sum_{(i,j,k)\in\overline{A}} (z_{ij}^k + kw_{ij}^k) \\ & \text{st.} \\ & \sum_{(i,j,k)\in\overline{A}} y_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} y_{li}^{(k-1)} = b_i, && \forall (i,k) \in \overline{N}, \\ & y_{ij}^k \in \{0,1\}, && \forall (i,j,k) \in \overline{A}, \\ & -My_{ij}^k \leq z_{ij}^k \leq c_{ij} + M(1-y_{ij}^k), && \forall (i,j,k) \in \overline{A}, \\ & c_{ij} - M(1-y_{ij}^k) \leq z_{ij}^k \leq My_{ij}^k, && \forall (i,j,k) \in \overline{A}, \\ & -My_{ij}^k \leq w_{ij}^k \leq \alpha_{ij} + M(1-y_{ij}^k), && \forall (i,j,k) \in \overline{A}, \\ & \alpha_{ij} - M(1-y_{ij}^k) \leq w_{ij}^k \leq My_{ij}^k, && \forall (i,j,k) \in \overline{A}, \\ & (c,\alpha) \in S, \end{aligned} \tag{PP-RTDSPP}$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$.

Henceforth we will refer to Problem $(PP-RTDSPP)$ as the *primal problem*. By solving the primal problem we obtain a worst-case scenario $(c^w(x), \alpha^w(x)) \in S$ for the solution $x \in \mathcal{P}$. Furthermore, by solving this primal problem we also obtain the maximum regret for x, $R(x)$, which defines an upper bound for the optimal objective value of the problem $(RTDSPP)$. A lower bound of this optimal value can be obtained through the *master problem*. In order to define it, let us suppose we have solved several primal problems $(PP-RTDSPP)$ for a set of feasible solutions x^τ, $\tau \in T$ and we have obtained the set of worst-case scenarios (c^τ, α^τ) for $\tau \in T$. In

what follows, for simplicity, we will represent by (c^τ, α^τ) the specific solution (worst-case scenario) of the primal problem for each iteration $\tau \in T$. The master problem associated to this set T has the following format

$$\begin{array}{rll} t^* = & \min & t \\ & \text{st.} & \\ & & x \in \mathcal{P}, \\ & & R(x, c^\tau, \alpha^\tau) \leq t, \forall \tau \in T. \end{array} \tag{3.17}$$

Given the set of scenarios $S_T = \{(c^\tau, \alpha^\tau) : \tau \in T\} \subset S$ obtained from the resolution of a family of primal problems, we have

$$\max_{(c,\alpha)\in S_T} R(x, c, \alpha) \leq \max_{(c,\alpha)\in S} R(x, c, \alpha)$$

that is

$$t^* = \min_{x\in\mathcal{P}} \max_{(c,\alpha)\in S_T} R(x, c, \alpha) \leq \min_{x\in\mathcal{P}} \max_{(c,\alpha)\in S} R(x, c, \alpha) = R^*. \tag{3.18}$$

Hence, solving the master problem allows us to bound from below the optimal objective R^* of the minmax regret TDSPP ($RTDSPP$).

In order to solve the master problem by a standard solver we must rewrite its formulation (3.17). First, note that, after solving the primal problem ($PP - RTDSPP$) corresponding to a feasible solution x^τ, $\tau \in T$, one obtains its worst-case scenario (c^τ, α^τ) and its worst-case adversary y^τ which is optimal for the deterministic problem under the scenario (c^τ, α^τ). Hence, according to the definition of the regret function in (3.5) we have

$$R(x, c^\tau, \alpha^\tau) = Z(x, c^\tau, \alpha^\tau) - Z(c^\tau, \alpha^\tau) = Z(x, c^\tau, \alpha^\tau) - Z(y^\tau, c^\tau, \alpha^\tau).$$

This identity allows us to rewrite the corresponding constraints of the master problem as linear inequalities on x using the expression of the function $Z(x, c, \alpha)$ given in ($DTDSPP$). Therefore, one has the following MILP formulation for the master problem

$$\begin{array}{rll} t^* = & \min & t \\ & \text{st.} & \\ & \displaystyle\sum_{(i,j,k)\in\overline{A}} x_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} x_{li}^{(k-1)} = b_i, & \forall (i,k) \in \overline{N}, \\ & \displaystyle\sum_{(i,j,k)\in\overline{A}} (x_{ij}^k - y_{ij}^{k\tau})(c_{ij}^\tau + k\alpha_{ij}^\tau) \leq t, & \forall \tau \in T, \\ & x_{ij}^k \in \{0,1\}, & \forall (i,j,k) \in \overline{A}, \end{array} \tag{MP-RTDSPP}$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$.

Let x^* be an optimal solution of Problem $(MP - RTDSPP)$ with objective value t^*. Now the process will continue computing the maximum regret of x^* by solving the corresponding primal problem $(PP - RTDSPP)$. Next, its worst-case scenario (c^*, α^*) is added to the set S_T after a new index is also added to T and the process is repeated. The optimality of the algorithm is guaranteed due to the finiteness of the set $\mathcal{P}$ and the following observation.

Proposition 10. *Let x^* be an optimal solution of the master problem associated to the set of indices T and (c^*, α^*) an optimal solution of its corresponding primal problem. Then x^* is an optimal solution for the minmax regret TDSPP $(RTDSPP)$ if there exists an index $\tau \in T$ for which $x^* = x^\tau$ or $(c^*, \alpha^*) = (c^\tau, \alpha^\tau)$.*

Proof. Let us first suppose that $x^* = x^\tau$ for a given $\tau \in T$. According to the formulation (3.17) we have

$$t^* \geq R(x^*, c^\tau, \alpha^\tau) = R(x^\tau, c^\tau, \alpha^\tau) = R(x^\tau),$$

where the last equality follows from the definition of the primal problem $(PP - RTDSPP)$.

Furthermore, we know by (3.18) that t^* is a lower bound for R^*, then

$$t^* \leq R^* \leq R(x^\tau),$$

hence $R(x^*) = R(x^\tau) = R^*$, that is, x^* is an optimal solution the minmax regret TDSPP $(RTDSPP)$.

Let us now suppose that $(c^*, \alpha^*) = (c^\tau, \alpha^\tau)$ for a given $\tau \in T$. In that case, considering the constraints of the problem (3.17), we have

$$R(x^*) \geq R^* \geq t^* \geq R(x^*, c^\tau, \alpha^\tau) = R(x^*, c^*, \alpha^*) = R(x^*).$$

Therefore, x^* solves the minmax regret TDSPP $(RTDSPP)$.

□ □

Using these results, we will propose three algorithms for solving the minmax regret TDSPP $(RTDSPP)$.

The first algorithm is based on a Benders Decomposition. A similar algorithm was originally applied to a Robust Shortest Path Problem (RSPP) in Montemanni and Gambardella (2005b). Here, we adapt it to our time-expanded network and in addition we initialize the algorithm with a Benders cut coming from a guaranteed 2-approximated solution.

Algorithm 3 Benders

procedure INITIALIZATION
  Choose $x^0 \in \mathcal{P}$ and solve the primal problem $(PP - RTDSPP)$ for $x = x^0$.
  Let (c^0, α^0) be an optimal solution for $(PP - RTDSPP)$. Take $T = \{0\}$, $R^+ := R(x^0)$, $\hat{x} = x^0$ and go to iteration $\nu = 1$.
procedure ITERATION $(\nu = 1, 2, \ldots)$
  Solve the master problem $(MP - RTDSPP)$ and let x^* be an optimal solution of such problem.
  Solve the primal problem $(PP - RTDSPP)$ for $x = x^*$. Let (c^*, α^*, y^*) be an optimal solution of such problem. If $R(x^*) < R^+$ take $R^+ := R(x^*)$ and $\hat{x} = x^*$.
  if $\exists \tau \in T$ verifying $x^* = x^\tau$ or $(c^*, \alpha^*) = (c^\tau, \alpha^\tau)$ **then**
    x^* is an optimal solution for the minmax regret problem $(RTDSPP)$. END.
  else if $R^+ - t^* \leq \epsilon$ **then**
    the solution $\hat{x}$ is an ϵ-approximation to the optimal solution of the problem $(RTDSPP)$. END.
  else
    take $x^\nu = x^*$, $(c^\nu, \alpha^\nu) = (c^*, \alpha^*)$, $T := T \cup \{\nu\}$, and go to iteration $\nu := \nu + 1$.

Remark 6. *In the first iteration of the Algorithm 3, $\nu = 1$, we can always take $x^* = y^0$, the worst-case adversary obtained in the initialization step when the primal problem $(PP - RTDSPP)$ is solved for $x = x^0$. This is true because $R(x, c^0, \alpha^0) = Z(x, c^0, \alpha^0) - Z(c^\tau, \alpha^0)$ hence, as in the first iteration it is solved the master problem $(MP - RTDSPP)$ for $T = \{0\}$, it is equivalent to minimize $Z(x, c^0, \alpha^0)$ with $x \in \mathcal{P}$, that is, y^0 is an optimal solution of the master problem according to the definition of the primal problem* (3.15) *and the formulation $(DTDSPP)$. Therefore, the statement of the first iteration of the algorithm could be modified according to the above considerations. However, we have preferred to state all the iterations in the same way in order to ease the description of the procedure.*

Remark 7. *Proposition 10 guarantees that the above Algorithm 3 is finite since the number of different paths in $\mathcal{P}$ and the number of extreme points of S, the candidates to be worst-case scenarios, are finite. Therefore, the algorithm will end with an optimal solution whenever $\epsilon = 0$. Otherwise, when $\epsilon > 0$ we can ensure the proposed solution $\hat{x}$, is an ϵ-approximation for Problem $(RTDSPP)$ since t^* is a lower bound of the optimal objective value R^* and $\hat{x}$ is the best solution generated during the execution of the algorithm. Therefore, one has*

$$0 \leq R(\hat{x}) - R^* \leq R^+ - t^* \leq \epsilon.$$

It is interesting to point out that the best lower bound t^* coincides in each iteration

with the optimal objective value of the corresponding master problem which will be greater than or equal to the above lower bound as can be seen from the definition of the master problem (3.17).

The second algorithm that we will analyze is based on a Branch and Cut approach using Benders cuts. To the best of our knowledge, this method has never been applied to a RSPP; however, a similar approach was used, for instance, by Pereira and Averbakh (2013) on the robust set covering problem. In addition, as mentioned in the introduction, we reinforce our Branch-and-Cut with a set of initial cuts from guaranteed 2-approximate solutions based on mid-point scenario solutions, as developed in Section 3.3.3.

In the next algorithm we propose, as referred in the first chapter, Benders cuts are used in a Branch and Cut scheme. First, our procedure starts using an incomplete description of the feasible set of the problem (3.17) and solving its linear relaxation. This initial description will be reinforced during the process by using separating Benders cuts in each subproblem generated with the branch-decision tree. Whenever an integer solution $\hat{x}$ of the master problem is found, with objective value $\hat{t}$, the primal problem is solved for the corresponding $\hat{x}$-solution. Let $(\hat{c}, \hat{\alpha}, \hat{y})$ be an optimal solution of such problem. If $R(\hat{x}) - \hat{t} \leq \epsilon$, $\hat{x}$ is a candidate to optimal solution and $\hat{t}$ a candidate to the optimal objective value, we keep them if they improve the incumbent solution. Otherwise, the solution is cut by introducing the constraint $\sum_{(i,j,k)\in\bar{A}}(x_{ij}^k - \hat{y}_{ij}^k)(\hat{c}_{ij} + k\hat{\alpha}_{ij}) \leq t^*$ to the master problem in the corresponding node of the branch-decision tree. The cut is propagated to all currently active nodes of the corresponding branch of the decision tree.

On the other hand, whenever a fractional solution of the master problem is found, with a depth lower than a prespecified level, one of the above cuts is also added to the master problem in order to separate that fractional solution.

Once the branch-decision tree has been explored, the incumbent solutions (if there were more than one) are optimal solutions of the problem. We encode the procedure in Algorithm 4

The advantages of this Branch and Cut approach are that the master problem is loaded only once and the cuts are added without restarting the branching tree after every cut is introduced. The disadvantage is that every cut to be added requires to solve a primal problem (and in some cases this number of primal problems to be solved is larger than in the Benders Algorithm 3).

Remark 8. *For the case in which the uncertainty set is a hypercube (Section 3.3.1) the primal problem is a shortest path problem. Therefore, these problems can be*

Algorithm 4 Branch and Cut

procedure INITIALIZATION
    Let $\mathbf{L}$ be the list of active problems and d_{max} be the maximum depth to get Benders cuts for fractional solutions.
    Set $\mathbf{L} := \{master\ problem\ (MP)\}$, $d_{MP} := 0$ and $t^* = +\infty$.
procedure REPEAT UNTIL $\mathbf{L}$ IS EMPTY
    Select and remove a problem $\hat{P}$ from $\mathbf{L}$ with associated depth $d_{\hat{P}}$.
    Solve the linear relaxation of $\hat{P}$. Let $\hat{x}$ be an optimal solution and $\hat{t}$ its corresponding objective value.
    if $\hat{x}$ is integer **then**
        solve the primal problem for $x = \hat{x}$. Let $(\hat{c}, \hat{\alpha}, \hat{y})$ be an optimal solution of such problem.
        if $R(\hat{x}) - \hat{t} \leq \epsilon$ **then**
            $\hat{x}$ is a candidate solution. If $\hat{t} \leq t^*$ then $t^* = \hat{t}$ and $x^* = \hat{x}$.
        else
            add the cut $\sum_{(i,j,k)\in\overline{A}}(x_{ij}^k - \hat{y}_{ij}^k)(\hat{c}_{ij} + k\hat{\alpha}_{ij}) \leq t$ to the problem $\hat{P}$.
            Add $\hat{P}$ to $\mathbf{L}$ with depth $d_{\hat{P}} = d_{\hat{P}} + 1$.
            Go to 1.
    else if $\hat{x}$ is not integer and $d_{\hat{P}} < d_{max}$ **then**
        solve the primal problem for $x = \hat{x}$. Let $(\hat{c}, \hat{\alpha}, \hat{y})$ be an optimal solution of such problem.
        Add the cut $\sum_{(i,j,k)\in\overline{A}}(x_{ij}^k - \hat{y}_{ij}^k)(\hat{c}_{ij} + k\hat{\alpha}_{ij}) \leq t$ to problem $\hat{P}$.
        Add $\hat{P}$ to $\mathbf{L}$ with depth $d_{\hat{P}} = d_{\hat{P}} + 1$.
        Go to 1.
    else
        Branch problem $\hat{P}$ into new problems with restricted feasible regions.
        Add these problems to $\mathbf{L}$ with associated depth $d_{\hat{P}} + 1$.
        Go to 1.

solved either directly by any solver as a linear program or with specific combinatorial algorithms for shortest path problems.

We propose a third algorithm which is a combination of the two previously described Algorithms 3 and 4. Observe that, if we include in the master problem all the Benders cuts corresponding to the extreme points of the polyhedron S, the master problem is a valid MILP formulation for problem $(RTDSPP)$. The main idea of this *combined* algorithm is to apply Benders Algorithm 3 until some threshold for the gap is achieved or until a fixed number of cuts are added to the master problem. By applying several iterations of the sequential Benders Algorithm 3, we obtain a master problem that provides a better description of problem $(RTDSPP)$. Once this tighter master problem is obtained the Branch and Cut Algorithm 4 is launched using as master problem the one obtained in the sequential Benders Algorithm 3 stopped with the above criteria. This is:

Algorithm 5 Combined Benders - Branch and Cut

1: **procedure** INITIALIZATION(Set max_c, the maximum numbers of cuts to be added to the master problem and ϵ the accuracy threshold.)
2: Apply Algorithm 3 until it reaches a precision ϵ or max_c cuts are added (what happens first).
3: Apply Algorithm 4 initializing $\mathbf{L}$ with the resulting master problem from step 1.

In what follows, we propose a feasible solution for the initialization step of the algorithms.

3.3.3 Initializing by 2-approximations

In the initialization of the proposed algorithms a solution x^0 is chosen arbitrarily. It would be interesting to use a solution as *close* as possible to the optimal one. In this sense, if the set of scenarios allows us to have a constant factor approximation such strategy could be used to initialize the algorithms. The following result provides a 2-approximation that can be used when the set of scenarios is a hypercube (see 1.1.1).

Proposition 11. *Let us consider* $S = \prod_{(i,j)\in A} [c_{ij}^-, c_{ij}^+] \times \prod_{(i,j)\in A} [\alpha_{ij}^-, \alpha_{ij}^+]$ *and the midpoint scenario* $(\overline{c}, \overline{\alpha})$*, that is,*

$$\overline{c}_{ij} = \frac{c_{ij}^- + c_{ij}^+}{2}, \quad \overline{\alpha}_{ij} = \frac{\alpha_{ij}^- + \alpha_{ij}^+}{2}, \forall (i,j) \in A,$$

then any optimal solution $\overline{x} \in \mathcal{P}$ *of the deterministic Problem* $(DTDSPP)$ *under the scenario* $(\overline{c}, \overline{\alpha})$ *is a 2-approximation for the minmax regret problem* $(RTDSPP)$.

Proof. Problem $(DTDSPP)$ can be rewritten equivalently as

$$
\begin{aligned}
Z(c,\alpha) = \quad & \min \quad Z(x,c,\alpha) := \sum_{(i,j)\in A} (c_{ij}y_{ij} + \alpha_{ij}w_{ij}) \\
& \text{st.} \\
& \sum_{(i,j,k)\in\overline{A}} x_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} x_{li}^{(k-1)} = b_i, \quad \forall (i,k) \in \overline{N}, \\
& \sum_{k:(i,j,k)\in\overline{A}} x_{ij}^k - y_{ij} = 0, \qquad \forall (i,j) \in A, \\
& \sum_{k:(i,j,k)\in\overline{A}} k x_{ij}^k - w_{ij} = 0, \qquad \forall (i,j) \in A, \\
& x_{ij}^k \geq 0, \qquad \forall (i,j,k) \in \overline{A},
\end{aligned}
\tag{3.19}
$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$.

The formulation (3.19) has a linear objective value and a compact feasible set, therefore (see, e.g. Conde (2010)) $\overline{x}$ is a 2-approximation for the minmax regret problem $(RTDSPP)$. □

The result presented in Proposition 11 can be extended to the searching problem with the specifications given in Section 3.3.1 and adding the existence of uncertainty intervals for the accessing costs.

Proposition 12. *Let us consider the searching problem in which $c_{ij} = nd_{ij} - r_j$ and $\alpha_{ij} = -d_{ij}$, where the coefficients d_{ij}, $(i,j) \in A$ and r_j, $j = 1, ..., n$ can take on any value from the uncertainty intervals $[d_{ij}^-, d_{ij}^+]$ and $[r_j^-, r_j^+]$ respectively. Let $(\overline{c}, \overline{\alpha})$ be the scenario defined as,*

$$
\overline{c}_{ij} = \frac{n(d_{ij}^- + d_{ij}^+) - (r_j^- + r_j^+)}{2}, \forall (i,j) \in A, \quad \overline{\alpha}_{ij} = \frac{-(d_{ij}^- + d_{ij}^+)}{2}, \forall j = 1, ..., n,
$$

then any optimal solution $\overline{x} \in \mathcal{P}$ of the deterministic problem $(DTDSPP)$ under the scenario $(\overline{c}, \overline{\alpha})$ is a 2-approximation for the minmax regret searching problem.

Proof. In this case, the formulation (3.19) is equivalent to

$$
\begin{aligned}
Z(c,\alpha) = \quad & \min \quad Z(x,c,\alpha) := \sum_{(i,j)\in A} (d_{ij} z_{ij} - r_j y_{ij}) \\
& \text{st.} \\
& \sum_{(i,j,k)\in\overline{A}} x_{ij}^k - \sum_{(l,i,k-1)\in\overline{A}} x_{li}^{(k-1)} = b_i, \quad \forall (i,k) \in \overline{N}, \\
& \sum_{k:(i,j,k)\in\overline{A}} x_{ij}^k - y_{ij} = 0, \qquad \forall (i,j) \in A, \\
& \sum_{k:(i,j,k)\in\overline{A}} k x_{ij}^k - w_{ij} = 0, \qquad \forall (i,j) \in A, \\
& n y_{ij} - w_{ij} - z_{ij} = 0, \qquad \forall (i,j) \in A, \\
& x_{ij}^k \geq 0, \qquad \forall (i,j,k) \in \overline{A},
\end{aligned}
\tag{3.20}
$$

where $b_0 = 1$, $b_{n+1} = -1$ and $b_i = 0$ for all $i \in N \setminus \{0, n+1\}$. Hence, we can apply the same argument as in the above proof.

□ □

3.4 Computational Experiment

In this last section we report some numerical experiments conducted to compare the different proposed algorithms to solve the minmax regret TDSPP under distinct types of uncertainty sets:

- either coefficients $c_{ij}, (i,j) \in A$ or parameters $\alpha_{ij}, (i,j) \in A$ are uncertain, but uncertainty intervals $[c_{ij}^-, c_{ij}^+]$ and $[\alpha_{ij}^-, \alpha_{ij}^+]$ are known, and the coefficients can vary independently from one another (Section 3.3.1).

- uncertainty affects simultaneously both coefficients c_{ij} and α_{ij}, $(i,j) \in A$, and not necessarily in an independent way (Section 3.3.2).

The computational experiments were carried out on a personal computer with Intel ® Core (TM) i7-4720HQ, 2.60GHz with 16384 MB RAM. The algorithms were implemented and solved by using Xpress Version: 8.0.

We generated, as in Chapter 2, instances of directed and acyclic square grid networks, following Fernández et al. (2014), adapting the SPGRID generator by Cherkassky et al. (1996). Recall that the $n = m \times m$ nodes of these graphs correspond to points on the plane with integer coordinates $[x,y]$, $1 \leq x \leq m$, $1 \leq y \leq m$. These points are connected *forward* by arcs of the form $([x,y],[x+1,y])$, $1 \leq x \leq m-1$, $1 \leq y \leq m$, *down* by arcs of the form $([x,y],[x,y-1])$, $1 \leq x \leq m$, $2 \leq y \leq m$ and *diagonally* by arcs of the form $([x,y],[x+1,y+1])$, $1 \leq x \leq m-1$, $1 \leq y \leq m-1$.

For the implementation of the algorithms proposed in the previous section we set the following parameters after experimenting with different values. In Algorithm 4, we set the depth for the fractional cuts to 3. In Algorithm 5, the *gap* and the *maximum number of iterations* of the sequential Benders Algorithm 3 were set to either 20% or 50% and 30 iterations, respectively. We report the best performance among these combinations. In order to perform a clean comparison of the different algorithms we disabled the presolving routines of Xpress solver and the remaining parameters were set to their default values. Our Branch and Cut code makes use of the framework provided by Xpress via callbacks.

3.4.1 Hypercube uncertainty set.

The computational experiment compares the algorithms proposed in Section 3.3.2 and the MILP formulation (3.13) for the TDSPP. In order to perform this comparison we generated instances of the general TDSPP with arc costs $c_{ij} + \alpha_{ij} posit_{ij}(x)$, assuming first that only coefficients $c_{ij}, \forall (i,j) \in A$, are uncertain varying in $[c_{ij}^-, c_{ij}^+]$ (Table 3.3), and then assuming also that coefficients α_{ij}, $\forall (i,j) \in A$, can take any value in the interval $[\alpha_{ij}^-, \alpha_{ij}^+]$ (Table 3.4). Finally, for the particular searching problem application, we generated instances with arc costs $c_{ij} + \alpha_{ij} posit_{ij}(x)$, with $c_{ij} = nd_{ij} - r_j$, $\alpha_{ij} = -d_{ij}$ and $r_j \in [r_j^-, r_j^+]$ (Table 3.5).

The lower and upper bounds of the uncertainty intervals for the general TDSPP were randomly generated following a uniform distribution in $(-1, 1)$ and in $(c_{ij}^-, 1)$ respectively. The TDSPP is very sensitive to variations in the parameters α_{ij} and the nature of the optimal solution changes significantly depending on the values of these parameters. For instance, if α_{ij} takes the value 0 for all the arcs, our problem becomes the standard Minmax Regret Shortest Path Problem. However, if it takes the same large enough value α for every arc, the optimal solution tends to have the smaller possible number of arcs. Taking this into account, each nominal α_{ij} was generated following a uniform distribution in $(0, \alpha_0)$, with α_0 equal to $\frac{1}{m}$ and $\frac{1}{3m}$ (experimentally we observed that the chosen values for the parameter α_0 generate *interesting* instances of the TDSPP for our grid networks). For the case in which the parameters α_{ij} are also assumed to be unknown, the intervals $[\alpha_{ij}^-, \alpha_{ij}^+]$ were built around the previous nominal α_{ij}, by generating a random value $\epsilon_{ij} \in [0, \alpha_{ij}]$ and defining the new extreme points of the uncertainty intervals as $\alpha_{ij}^- = \alpha_{ij} - \epsilon_{ij}$ and $\alpha_{ij}^+ = \alpha_{ij} + \epsilon_{ij}$ $\forall (i,j) \in A$.

The accessing costs d_{ij} for the searching problem were randomly generated from a uniform distribution in $(0, 1)$. The bounds of the uncertainty intervals for the parameters r_j were generated with the aim that the reward that can be obtained by visiting a node were larger than the corresponding accessing cost for some nodes, this is, $r_j \geq (n - posit_{ij})d_{ij}$ for some $(i,j) \in A$ (note that the cost associated to each

arc (i,j) is $(n-posit_{ij})d_{ij}-r_j)$. Following this rationale, r_j^- and r_j^+ were randomly generated from the interval $\left(p_j^{mid}, m^2\bar{d}_j + \frac{L_j}{q}\right)$ for every j, with $q=4$, where p_j^{mid} and L_j denote the mid-point and the length of the interval $[(m^2-3m+3)\bar{d}_j, m^2\bar{d}_j]$, respectively, and $\bar{d}_j$ is the average accessing cost of the arcs ending at node j. The structure of these intervals was determined by the fact that $(n-posit_{ij})d_{ij} \in [(m^2-3m+3)d_{ij}, m^2 d_{ij}]$ depending on the position of arc (i,j) for our particular grid graphs.

Ten different instances of each problem were generated for $n=100$, 169, 225, 324, 400. We show in Tables 3.3, 3.4 and 3.5 the number of problems (out of 10) solved to optimality for each n, in less than 7200 seconds (# opt), the average gaps and the average computational times (CPU) for the three algorithms and the MILP formulation. In those cases in which the execution process is stopped before an optimal solution is found we report, in our computational experiments, the CPU time limit of 7200 seconds. Table 3.4 does not report results for $n=400$ since already for $n=324$, most of the instances could not be solved to optimality by the algorithms and even, the exact formulation for these instances could not be charged by the solver.

We observe in Tables 3.3, 3.4 and 3.5 that the fastest numerical procedure is the Combined Algorithm 5. For example, in Table 3.3, when $\alpha_0 = \frac{1}{m}$ and $n=225$ the average CPU time of the Benders Algorithm 3 is almost twice the average CPU time of the combined one, and in Table 3.4, in the case in which $\alpha_0 = \frac{1}{3m}$ and $n=225$, the Exact Formulation and Algorithms 3 and 4 spent, on average, more than 5300 seconds to solve the instances whereas the Combined Algorithm 5 spent 3305.4 seconds. Furthermore, it can be seen in Table 3.3 that the Branch and Cut Algorithm 4 is also faster than both, the Benders Algorithm 3 and the exact formulation, for most of the instances of the general TDSPP with only uncertainty on the α_{ij} parameters. However, as it can be seen in Table 3.5, Benders Algorithm 3 works faster for the searching problem instances of sizes $100, 169$ and 225. This performance is due to the fact that Benders Algorithm 3 performs better when the solution can be found with a few Benders cuts, and in this experiment we observed that the number of required cuts in the instances solved to complete Table 3.5 was *small*. Nevertheless, when the number of cuts that need to be added increases, the performance of Benders Algorithm 3 gets worse due to the increase of the number of master problems that have to be solved.

Tables 3.3, 3.4 and 3.5 also show that the Combined Algorithm 5 is able to solve instances that cannot be solved using either the exact formulation or Algorithms 3 and 4 within the time limit of 7200 seconds. In fact, for all the cases, the Combined Algorithm 5 is the one that solved the major number of instances. For example when

the parameter α_0 takes the value $\frac{1}{3m}$ and both coefficients c_{ij} and α_{ij} are uncertain (Table 3.4) the Combined Algorithm 5 could solve 9 out of 10 instances for the size 225, whereas the exact formulation and Algorithms 3 and 4 could only solve 0, 5 and 4 instances, respectively. In the same way, when n increases to 324, and $\alpha_0 = \frac{1}{m}$, the Combined Algorithm 5 could solve 5 out of 10 instances, whereas Algorithms 3 and 4 could only solve 1 instance, and the exact formulation could not even be charged by the solver. Furthermore, in those problems where only the coefficients c_{ij} are uncertain (Table 3.3), for $\alpha_0 = \frac{1}{m}$, the Combined and the Benders algorithms were able to solve 8 and 4 out of 10 instances, respectively, when $n = 324$ within the bound of 7200 seconds. Similarly, we can observe in Table 3.5 that when we consider the instances for graphs with 400 nodes, the Combined algorithm could solve all the instances while the Branch and Cut algorithm was able to solve only 7 out of 10.

Finally, from Tables 3.3 and 3.4 we can also conclude that the smallest average gaps, for the general TDSSP, are achieved by the Combined Algorithm 5.

		Exact Formulation			Benders Algor. 1			Branch and Cut Algor. 2			Combined Algor. 3		
α_0	**n**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**
$1/m$	100	10	15.2	0.0	10	30.1	0.0	10	22.2	0.0	10	22.4	0.0
$1/m$	169	10	293.4	0.0	10	1102.5	0.0	10	270.2	0.0	10	280.5	0.0
$1/m$	225	10	727.9	0.0	10	1161.1	0.0	10	950.0	0.0	10	579.0	0.0
$1/m$	324	8	3987.1	2.7	4	5785.5	3.1	7	3925.5	4.8	8	3981.5	2.9
$1/m$	400	3	6847.2	24.1	2	6895.1	9.0	4	5496.2	9.4	5	5949.1	7.6
$1/3m$	100	10	64.3	0.0	10	44.0	0.0	10	38.9	0.0	10	41.2	0.0
$1/3m$	169	10	443.9	0.0	10	399.2	0.0	10	281.3	0.0	10	249.2	0.0
$1/3m$	225	8	3649.1	2.3	8	3146.2	2.4	8	2952.7	2.3	9	2233.7	1.6
$1/3m$	324	1	6808.6	30.1	1	7167.7	10.7	1	7011.3	13.0	4	6888.1	8.8
$1/3m$	400	1	7122.7	47.2	1	7108.9	18.4	1	6798.5	18.6	2	6596.7	11.5

Table 3.3: Computational results for the general minmax regret TDSPP with hypercube uncertainty sets for parameters c_{ij}.

		Exact Formulation			Benders Algor. 1			Branch and Cut Algor. 2			Combined Algor. 3		
α_0	**n**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**
1/m	100	10	108.4	0.0	10	120.5	0.0	10	85.5	0.0	10	85.1	0.0
1/m	169	10	1159.8	0.0	10	1350.6	0.0	9	2017.2	2.0	10	1123.4	0.0
1/m	225	9	3622.8	13.1	9	2698.4	0.0	8	2688.3	1.8	10	1474.2	0.0
1/m	324	**	**	**	1	6875.8	9.0	1	6829.6	5	5	6873.0	5.4
1/3m	100	10	187.2	0.0	10	104.3	0.5	10	138.7	0.0	10	100.2	0.0
1/3m	169	10	2358.9	0.0	10	898.9	0.1	10	1258.8	0.0	10	808.5	0.0
1/3m	225	0	7890.2	281.6	5	5303.2	5.5	4	5931.8	10.1	9	3305.4	0.0
1/3m	324	**	**	**	0	7200.0	13.3	0	7200.0	14.3	2	7084.3	11.2

Table 3.4: Computational results for the general minmax regret TDSPP with hypercube uncertainty sets for parameters c_{ij} and α_{ij}. ** The solver returns an out of memory code.

	Exact Formulation			Benders Algor. 1			Branch and Cut Algor. 2			Combined Algor. 3		
n	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**
100	10	40.6	0.0	10	9.9	0.0	10	18.3	0.0	10	10.6	0.0
169	10	273.7	0.0	10	24.2	0.0	10	51.4	0.0	10	24.2	0.0
225	9	1913.4	10.0	10	296.7	0.0	10	505.4	0.0	10	354.1	0.0
324	6	4183.3	44.2	10	584.7	0.0	10	1543.7	0.0	10	551.9	0.0
400	8	4703.4	22.2	10	802.0	0.0	7	3253.2	3.6	10	733.3	0.0

Table 3.5: Computational results for the minmax regret searching problem with hypercube uncertainty sets.

3.4.2 Polyhedral uncertainty set.

To end this computational experiment we are going to consider the minmax regret problem (RTDSPP) under a polyhedral uncertainty set of scenarios S. The instances of this optimization model will be generated according to the searching problem of Section 3.1.1 in which it will be allowed that the uncertainty affects simultaneously to both types of coefficients: the accessing costs d_{ij} and the rewards r_j. Each one of these coefficients is supposed to take on any value in their corresponding uncertainty interval, $d_{ij} \in [d_{ij}^-, d_{ij}^+]$ and $r_j \in [r_j^-, r_j^+]$ $\forall (i,j) \in A$. This uncertainty translated in terms of the coefficients c and α gives rise to the following uncertainty polyhedron $S = \{(c,\alpha) : -d_{ij}^+ \leq \alpha_{ij} \leq -d_{ij}^-,\ -r_j^+ \leq c_{ij} + n\alpha_{ij} \leq -r_j^-\ ,\ \forall (i,j) \in A\}$. Observe that S is a bounded set and the constant M, defined in (3.16), can be easily obtained as

$$\begin{aligned} \alpha_{max} &= \max\{|d_{ij}^-|, |d_{ij}^+| : (i,j) \in A\}, \\ c_{max} &= \max\{|n\alpha_{max} - r_j^-|, |n\alpha_{max} + r_j^+| : j = 1, \ldots, n\}, \\ M &= \max\{\alpha_{max}, c_{max}\}. \end{aligned}$$

Ten different instances of each problem were generated for $n = 16, 25, 36, 49$. Note that the number of nodes, n, of the graph has been reduced drastically with respect to the instances solved in the previous subsection due to that the difficulty of the problems becomes much higher. For each instance, the vectors d^- and d^+ were randomly generated from a uniform distribution in $(0,1)$, and the uncertainty interval bounds for the rewards, r_j^- and r_j^+, were randomly generated in $\left(p_j^{mid}, m^2\bar{d}_j + \dfrac{L_j}{q}\right)$ for every j, with $q = 4$, as in the previous subsection.

We report in Table 3.6 the number of problems (out of 10) solved to optimality for each n in less than 7200 seconds (# opt), the average gaps and the average computational times (CPU) for the three algorithms. The initialization of these algorithms was made using the cost scenario given by the mid-points of the uncertainty intervals of the coefficients d_{ij} and r_j according to Proposition 12.

Table 3.6 shows that Benders Algorithm 3 is always faster than the other two algorithms. For instance, the average CPU time for 25 nodes instances is 91.9 seconds whereas it is larger than 370 seconds for the other algorithms. The reason is that the primal problem (PP-RTDSPP) is an unstructured general LP problem, and the Branch and Cut algorithm has to solve one of these problems in each node of the branching tree. In contrast to the hypercube uncertainty case, when we consider polyhedral uncertainty sets, the saving obtained by the Branch and Cut algorithm by not solving several master problems does not compensate the number of primal problems that need to be solved.

Furthermore, Table 3.6 also shows that in spite of the fact that Benders Algorithm 3 is faster, Combined Algorithm 5 is able to solve optimally the same number of problems for $n = 49$, 6 out of 10, with a smaller average gap, that is, near half of the average gap incurred by Algorithm 3.

	Benders Algor. 1			**Branch and Cut Algor. 2**			**Combined Algor. 3**		
n	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**	**# opt**	**CPU**	**gap**
16	10	19.1	0.0	10	91.7	0.0	10	67.2	0.0
25	10	91.9	0.0	10	496.7	0.0	10	379.3	0.0
36	10	557.2	0.0	10	2673.6	0.0	10	1835.9	0.0
49	6	4770.5	6.3	1	7028.2	20.7	6	7016.5	3.2

Table 3.6: Computational results for the minmax regret searching problem with polyhedral uncertainty sets.

3.5 Conclusions

In this chapter we dealt with a SPP model with costs dependent on the position occupied by the corresponding arcs in the path, the time-dependent SPP. The problem has been studied when the cost parameters are unknown using the minmax regret paradigm. Two methodologies have been developed, the first one consists on solving a MILP formulation, the second one applies Benders cuts in order to have an approximation to the optimal solution.

The MILP formulation works for the TDSPP when hypercube uncertainty sets are considered. The second methodology is more general. It allows us to consider the minmax regret time-dependent SPP problem under a polyhedral set of possible cost scenarios. Moreover, including in the primal and master problems the constraint that the desired path passes through every node exactly once, the same framework can also be applied to the minmax regret time-dependent TSP problem. This opens a wider set of real applications in timetabling and scheduling, as mentioned in the introduction. Using linear constraints to model dependency relationships amongst the unknown parameters allows us to face new and more realistic situations.

Three algorithms have been proposed to solve the problems. These algorithms make use of Benders cuts that, with some variants, are added to a master problem whose solutions become closer to the optimal solution over successive iterations. As it has been reported in the numerical experiments, the performance of these iterative algorithms has been improved by starting the procedure using cuts associated to constant factor approximations. In this chapter we have used 2-approximations obtained, following the propositions 11 and 12, from an optimal solution under the mid-point scenario. However, these approximations are only defined for the cases in which every uncertain cost varies independently of each other in its corresponding

uncertainty interval. It would be interesting to study the possible extension of these results to more general uncertainty sets Conde (2012). This could give rise to a research line consisting on the adaptation of these schemes to improve the Benders cuts algorithm when applied to uncertainty sets where the 2-approximations obtained from mid-point scenarios cannot be used.

Another possible improvement of the algorithms proposed in this chapter could come from the application of size reduction techniques to the time-expanded network. The size of this network can easily grow in actual applications limiting the efficiency of the algorithms, hence it would be a good idea having some type of preprocessing which prunes the network before applying the numerical approach. For instance, Karasan et al. (2001), and more recently Catanzaro et al. (2011), provided sufficient conditions for a node and an arc to be always or never in an optimal solution of the minmax regret SPP. It seems interesting to analyze the possible extension of sufficient conditions like the ones proposed in those papers to the minmax regret time-dependent models in order to develop variable fixing strategies (pegging tests) to reduce the size of the network.

Chapter 4

Minmax regret combinatorial optimization problems with investments

In this chapter, we propose new minmax regret optimization models in a system with uncertain parameters, in which it is allowed to make investments before a minmax regret solution is implemented in order to modify the source or the nature of the existing uncertainty. Therefore, it is allowed to spend resources in order to change the basic cost structure of the system and take advantage of the modified system to find a robust solution. Some properties of this model allow us to have proper Mathematical Programming formulations that can be solved by standard optimization packages. As a practical application, we consider the shortest path problem in a network in which it is possible to modify the uncertainty intervals for the arc costs by investing in the system. We also give an approximate algorithm and generalize some existing results on constant factor approximations.

4.1 Introduction

Most of the minmax regret optimization models considered in the literature (see e.g. Chapter 1 and the references therein) rule out the fact of having some kind of control over the set of unknown parameters. However, the source of uncertainty could be modified or the degree of knowledge about the possible scenarios could change as a consequence of particular actions carried out in the system. For instance, as pointed out in the introduction, if we consider a problem in which the demand of a customer is modeled as an uncertain parameter and we assume that this parameter can take on any value from an interval fixed beforehand, then the length of this uncertainty interval can be reduced by performing a prospective study and obtaining some extra information about its behavior.

It can be logically assumed that this kind of actions that modify the uncertainty set of possible scenarios involves certain costs in terms of a set of limited resources. On the one hand, these costs can be considered as an investment in the system with the purpose of gaining insight into the mechanism involved in its behavior and obtaining, in this way, more information about the uncertain elements (costs, capacities, lengths, times, etc.). On the other hand, the investment can also have a different purpose, the available resources could be spent to modify the nature of the system itself. For example, if we consider a communication network with uncertain parameters, like transfer speed or capacity of the connections, one can invest in order to increase this transfer speed or the capacity of a given connection or even invest in the construction of new connections. In all these cases we are investing in the system in order to modify the range of possible values of the uncertain parameters.

Then, the problem will consist in determining how to invest the available resources in order to obtain a robust solution *as close as possible* to the optimal one under whatever scenario that may occur in the transformed system. Therefore, the

purpose of the investments is to protect the decision-maker, in some way, against the uncertainty of implementing a feasible solution without the knowledge of the effects that the behavior of the system could have on it.

In the next section, we will assume some rationality principles for the optimization model in order to avoid undesirable effects of the proposed solution. For instance, if we allow the possibility of investing in order to deteriorate the behavior of the system, the optimization model could propose investing to get a poor performance of the system for those scenarios under which it initially worked fine, which would be nonsense. In this way, by making more homogeneous this (bad) behavior of the system under different scenarios we could find a solution with a small regret whatever scenario may occur. In order to avoid this undesirable effect it will only be allowed those investments that improve in some way the performance of the system. For example, if the interval representing the range of possible costs of a given item or action is reduced then the behavior of the whole system will be improved because it is more predictable. In the limit, we could think that, if we had enough resources, we could invest them in the system in order to eliminate all the existing uncertainty, reducing to zero the maximum regret incurred by an optimal solution under the only possible scenario. The performance of the system, in terms of its cost structure, will also be improved if the length of an interval of possible costs remains constant but its lower and upper bounds simultaneously decrease.

In what follows, we will consider two different types of decision variables of the mathematical formulation of the model; $t \in T$ modeling the feasible investments that will determine the set of possible scenarios and $x \in X$ defining the feasible actions that can be implemented in the system under any of the possible scenarios. We will seek a feasible solution minimizing the maximum regret in the resulting system after the investment has been carried out. Logically, the computational complexity of the resulting problems will be harder than that of its corresponding counterparts in which no investments are allowed, that is, it will be, almost without exception, $\mathcal{NP}$-hard for most of the combinatorial optimization problems considered in the literature (see e.g. Section 1.1.1 and the references therein). This high computational complexity underlines the importance of the study of approximate solutions and, if it is possible, the construction of a bound for the incurred error. After obtaining some properties of the general model, a possible generalization of existing results about constant factor approximations under uncertainty intervals will be proposed. In this chapter, the *Shortest Path Problem* (SPP) will be used as the reference optimization problem where the different definitions and formulations will be shown, nonetheless, as it will be pointed out later, the methodology and results we present in this chapter can be applied to other combinatorial optimization problems. Finally, a computational study for the minmax regret SPP in which it is allowed a set of

possible investments to modify the cost structure is conducted in order to compare the exact solution obtained by means of a professional optimization software with respect to the approximation proposed in this chapter.

4.2 Model description

Let us consider the *deterministic combinatorial optimization problem* defined in Section 1.1.1, that is, let A be a finite set of elements and X be a *feasible set* of solutions identified as subsets of A, that is, every $x \in X$ is a 0-1 (*characteristic*) vector with as many components as the number of elements in A, $x = (x_a)_{a \in A}$, where $x_a = 1$ if and only if a belongs to the subset represented by x. We assume that every element $a \in A$ has an associated weight c_a (cost, length, time, etc.). Recall that this problem is given by

$$\begin{array}{ll} \min & \sum_{a \in A} c_a x_a \\ \text{subject to} & \\ & x \in X. \end{array}$$

As said in Chapter 1, this formulation encompasses a wide range of optimization problems, but in many practical applications, the weights c_a are only partially known. This situation can be modeled by assuming that an interval of possible values of each weight is known in advance, that is, $c_a \in [c_a^-, c_a^+]$, $a \in A$. Additionally, it could be interesting and realistic to consider that these *uncertainty intervals* are not invariable, but they can be modified by making some kind of investment in the system. Specifically, we identify by a compact set T these feasible investments in the system and for each investment $t \in T$, an interval $[c_a^-(t), c_a^+(t)]$ of possible weights for c_a, regardless of the weights of the other elements.

In what follows we will study how to invest in the system in order to find a feasible solution with a performance as close as possible to the optimal one under each possible scenario. If we denote by S_t the set of possible scenarios that may occur after a given investment $t \in T$ has been carried out in the system, that is, the cartesian product of all the uncertainty intervals and, by c_a^s, the weight associated to the element $a \in A$ under the scenario $s \in S_t$, we can state the *minmax regret combinatorial optimization problem with Investments* as

$$R^* = \min_{t \in T} \min_{x \in X} \max_{s \in S_t} \left[R(x,t,s) := \left\{ \sum_{a \in A} c_a^s x_a - \min_{y \in X} \sum_{a \in A} c_a^s y_a \right\} \right]. \qquad \text{(RCI)}$$

In this case, the value $R(x,t,s)$ is the regret of x under the scenario $s \in S_t$.

The structure of the feasible set X, where each solution is a $0-1$ vector, and the independent variation supposed for the weights allow us to compute the maximum regret for a solution x by using the following property (see e.g. Section 1.1.1)

Proposition 13. *The regret of a solution $x \in X$ for a given investment $t \in T$ is maximized under the scenario $s(x,t)$ defined as follows:*

$$c_a^{s(x,t)} = \begin{cases} c_a^+(t) & if\ x_a = 1, \\ c_a^-(t) & if\ x_a = 0. \end{cases} \qquad a \in A.$$

Hence, as done before, having a closed-form of the *worst-case* scenario $s(x,t)$ allows a more compact rewriting of the problem (RCI):

$$\min_{t\in T, x\in X} \left\{ \sum_{a\in A} c_a^+(t)x_a - \min_{y\in X} \sum_{a\in A} \left[(c_a^+(t) - c_a^-(t))x_a + c_a^-(t) \right] y_a \right\}. \tag{4.1}$$

Henceforth, some additional conditions on the sets X and T will be imposed in order to obtain a proper Mathematical Programming formulation of (4.1) that can be solved by a standard optimization software. Furthermore, it will be considered that the bounds of the uncertainty intervals (weight/cost bound functions) have certain monotonic properties with respect to the amount of invested resources. As it was mentioned before, the investment of limited resources in the system should guarantee, in some sense, a *better* running in terms of costs or in terms of a reduction of the existing uncertainty. We will also consider that these weight bound functions have linear or piecewise linear structure in order to make easier the numerical resolution of the resulting formulation.

Total unimodularity of the feasible set of actions Hereafter it will be assumed that the structure of the feasible set is given by

$$X = \{x \in \{0,1\}^{|A|} : Mx = b\},$$

where M is a totally unimodular matrix and b is an integer n-vector. Under this condition the binary constrains on the decision variables can be replaced by bounds on continuous variables, $0 \leq x_a \leq 1$, for every element $a \in A$ and the inner sub-problem of the formulation (4.1) can be solved as a linear optimization problem for a given investment $t \in T$. Furthermore, the linear structure of this problem allows us to utilize duality in order to obtain a more suitable formulation of the problem from the optimization viewpoint (see Section 1.1.1).

Duality In order to rewrite the problem (RCI) in a format that can be recognized by standard optimization packages we will use linear duality in the inner problem that obtain the minimum weight under the scenario $s \in S_t$ for a given investment $t \in T$, that is,

$$\min_{y \in X} \sum_{a \in A} c_a^s y_a. \tag{4.2}$$

As previously said in the first chapter, this technique has been used in other optimization problems where the deterministic counterpart is linear. In the considered case, the total unimodularity of the matrix M defining the set X guarantees that this set can be replaced by its continuous relaxation in (4.2). Furthermore, the existence of a closed-form of the worst-case scenario for any possible solution (Proposition 13) allows us to eliminate one of the inner optimization problem of the formulation (RCI) where the regret is maximized. After using linear duality, we obtain an equivalent formulation of the problem (RCI), already known for fixed $t \in T$ (see e.g. Aissi et al. (2009)):

$$\begin{aligned}
&\min \sum_{a \in A} c_a^+(t) x_a - \sum_{i=1}^{n} b_i u_i \\
&\text{subject to} \\
&\quad Mx = b, \\
&\quad \sum_{k=1}^{n} m_k^a u_k \leq c_a^-(t) + (c_a^+(t) - c_a^-(t)) x_a, \ \forall a \in A, \\
&\quad x_a \in \{0,1\}, \quad \forall a \in A, \\
&\quad t \in T,
\end{aligned} \tag{RCI-D}$$

where m_k^a is the k-th component of the column vector of the matrix M associated to the variable x_a for each element $a \in A$.

Polyhedron of investments We will consider that the set of possible investments is a bounded polyhedron. The variable t_a^k, with $k \in K$ and $a \in A$, represents the amount of the resource k which is invested in the element a. The total amount of each resource will be assumed to be limited. In addition to these constraints, the polyhedron T could include some other type of linear constraints. For example, we could consider blocking conditions that avoid investing in a given subset of elements A_1 or limit such an investment if it has not been invested anything in another subset of elements A_2, that is, constraints like

$$\sum_{a \in A_1} t_a^k \leq \lambda_k(A_1, A_2) + \mu_k(A_1, A_2) \sum_{a \in A_2} t_a^k,$$

for given constants $\lambda_k(A_1, A_2)$ and $\mu_k(A_1, A_2)$. In other cases, it could be necessary to add some constraints to the set T in order to guarantee that the uncertainty intervals of weights are not empty, that is, $c_a^-(t) \leq c_a^+(t)\ \forall t \in T, a \in A$. If these constraints can be written in linear form, the polyhedral structure of T will be preserved.

Monotonic weight/cost bound functions In the particular examples considered in the following sections we will only allow investments that improve in some sense the performance of the system. This condition may not be strictly necessary in order to have a structured problem that can be solved, however it is convenient in order to provide useful and meaningful solutions for the system. In our model the bounds $c_a^{\pm}(t)$ are considered piecewise linear functions and the upper bound $c_a^+(t)$ will also be assumed to be a component-wise non-increasing function. Additionally, for each element $a \in A$, either the lower bound $c_a^-(t)$ or the difference between both bounds, $c_a^+(t) - c_a^-(t)$, is also assumed to be a component-wise non-increasing function. By assuming this property in the cost bound functions, the investment in the system achieves the goal of improving the weights or reducing the existing uncertainty about them.

In the following subsection, we will write the formulation $(RCI - D)$ as an equivalent MILP for some particular choices of the weight bound functions $c_a^{\pm}(t)$, $a \in A$.

4.3 Model formulation

Problem $(RCI - D)$ is a general nonlinear mixed integer optimization problem, hence, in order to design a numerical algorithm to solve it, it is necessary to impose additional conditions on the cost bound functions. The goal is to obtain a reformulation of this problem as a MILP. Hereafter we will consider a specific structure of the finite set of elements A, it will be defined as the set of arcs of a given network $G = (V, A)$ where $V = \{1, \ldots, n\}$ is the set of nodes. Now, a generic element $a \in A$ will be replaced by an ordered pair of nodes $a = (i, j)$, $i, j \in V$ and the weight $c_a = c_{ij}$ will define a cost (time or length) associated to the corresponding arc.

In this section, we will see how to linearize the resulting problem after a particular shape for each cost bound function has been assumed. In order to do it, a set of auxiliary variables will be defined as products of the original ones. Although this change of variables can be made for general sets of investments T, for the sake of simplicity we will restrict ourselves to consider a set with just one type of resource. By the variable $t_{ij} \geq 0$ we represent the amount of the given resource invested in the arc $(i, j) \in A$, with a total amount of available resource of 1 unit that can not be exceeded. Therefore, we consider an initial set of possible investments as the

following one

$$T = \left\{ (t_{ij})_{(i,j)\in A} : \sum_{(i,j)\in A} t_{ij} \leq 1,\, t_{ij} \geq 0, \forall (i,j) \in A \right\}, \tag{4.3}$$

having in mind that it is possible we need to add some additional constraints to this set in order to guarantee that the cost bound functions will define nonempty uncertainty intervals for the corresponding costs.

Despite of the fact that the new mathematical formulation considered here can be applied to any problem with totally unimodular feasible set, it will be adapted in this section to a specific set of feasible solutions, namely the set of paths between the nodes 1 and n in the network G. Hence, we will consider in this section the particular model of the Minmax Regret or Robust Shortest Path Problem with Investments (RSPPI). The reason is not other than illustrating the minmax regret with investments model by means of a particular problem whose resulting transformed formulation will be used to conduct a further computational study to compare exact and approximate methods on a specific problem. Therefore, the general formulation $(RCI-D)$ can be now written as

$$\begin{aligned}
&\min \sum_{(i,j)\in A} c_{ij}^{+}(t) x_{ij} - u_n \\
&\text{subject to} \\
&\quad -\sum_{(i,j)\in A} x_{ij} + \sum_{(j,k)\in A} x_{jk} = b_j, \quad j = 1,2,...,n, \\
&\quad u_j \leq u_i + c_{ij}^{-}(t) + (c_{ij}^{+}(t) - c_{ij}^{-}(t)) x_{ij}, \quad \forall (i,j) \in A, \\
&\quad u_1 = 0, \\
&\quad x_{ij} \in \{0,1\}, \quad \forall (i,j) \in A, \\
&\quad t \in T,
\end{aligned} \tag{RSPPI}$$

where $b_1 = 1,\ b_n = -1,\ b_j = 0\ \forall j \in V \setminus \{1,n\}$,

$$x_{ij} = \begin{cases} 1 & \text{if we include the arc } (i,j) \text{ in the path,} \\ 0 & \text{otherwise,} \end{cases} \qquad \forall (i,j) \in A,$$

and $u_i,\ i = 1,...,n$ are dual variables.

4.3.1 Linear cost bound functions

The simplest shape of a cost bound function is the one describing a variation in cost proportional to the investment carried out, that is,

$$\begin{aligned} c_{ij}^{-}(t) &= c_{ij}^{-} - \alpha_{ij}^{-} t_{ij}, \\ c_{ij}^{+}(t) &= c_{ij}^{+} - \alpha_{ij}^{+} t_{ij}, \end{aligned}$$

where $\alpha_{ij}^{-}, \alpha_{ij}^{+}, c_{ij}^{-}, c_{ij}^{+} \in \mathbb{R}^{+}$ and $c_{ij}^{-}(t) \leq c_{ij}^{+}(t)\ \forall t \in T$.

By replacing these cost functions in the formulation (RSPPI) one has the following quadratic problem,

$$\begin{aligned} &\min \sum_{(i,j)\in A} (c_{ij}^{+} - \alpha_{ij}^{+} t_{ij}) x_{ij} - u_n \\ &\text{subject to} \\ &- \sum_{(i,j)\in A} x_{ij} + \sum_{(j,k)\in A} x_{jk} = b_j, \quad j = 1, 2, ..., n, \\ &u_j \leq u_i + c_{ij}^{-} - \alpha_{ij}^{-} t_{ij} + (c_{ij}^{+} - \alpha_{ij}^{+} t_{ij} - (c_{ij}^{-} - \alpha_{ij}^{-} t_{ij})) x_{ij}, \quad \forall (i,j) \in A, \\ &\sum_{(i,j)\in A} t_{ij} \leq 1, \\ &u_1 = 0, \\ &x_{ij} \in \{0,1\}, \quad \forall (i,j) \in A, \\ &t_{ij} \geq 0, \quad \forall (i,j) \in A, \end{aligned} \tag{4.4}$$

where $b_1 = 1,\ b_n = -1,\ b_j = 0\ \forall j \in V \setminus \{1, n\}$.

In order to linearize this formulation we define a new set of variables given by

$$p_{ij} = \begin{cases} t_{ij} & \text{if } x_{ij} = 1, \\ 0 & \text{otherwise}, \end{cases} \qquad \forall (i,j) \in A. \tag{4.5}$$

Therefore, adding the following set of constraints to the previous formulation to ensure that p_{ij} is the wished product

$$\begin{aligned} &p_{ij} \leq x_{ij}, \\ &p_{ij} \leq t_{ij}, \\ &p_{ij} \geq t_{ij} + x_{ij} - 1, \\ &p_{ij} \geq 0 \end{aligned}$$

we can replace each product $t_{ij} x_{ij}$ by p_{ij} obtaining, in this way, a MILP formulation

for the problem (RCI).

Observe that if we consider cost bound functions with the same slope for each arc, that is,

$$c_{ij}^-(t) = c_{ij}^- - \alpha_{ij} t_{ij},\ c_{ij}^+(t) = c_{ij}^+ - \alpha_{ij} t_{ij},\quad \forall (i,j) \in A,$$

the length of the uncertainty cost intervals remains constant for every $t \in T$.

Using these functions we obtain a simpler Mixed Integer Quadratic Formulation for the problem (RCI) in which only the objective is a quadratic function, that is,

$$\begin{aligned}
&\min \sum_{(i,j)\in A} (c_{ij}^+ - \alpha_{ij} t_{ij}) x_{ij} - u_n \\
&\text{subject to} \\
&\quad - \sum_{(i,j)\in A} x_{ij} + \sum_{(i,j)\in A} x_{jk} = b_j, \quad j = 1, 2, ..., n, \\
&\quad u_j \le u_i + c_{ij}^- - \alpha_{ij} t_{ij} + (c_{ij}^+ - c_{ij}^-) x_{ij}, \quad \forall (i,j) \in A, \\
&\quad \sum_{(i,j)\in A} t_{ij} \le 1, \\
&\quad u_1 = 0, \\
&\quad x_{ij} \in \{0,1\}, \quad \forall (i,j) \in A, \\
&\quad t_{ij} \ge 0, \quad \forall (i,j) \in A.
\end{aligned} \tag{4.6}$$

Some commercial solvers allow to solve this problem directly (like CPLEX), in any case it can be linearized following the same procedure as before.

4.3.2 Piecewise linear cost bound functions

We first discuss the case in which the lower bound for the cost of each arc is defined by the function

$$c_{ij}^-(t) = \max\{c_{ij}^- - \alpha_{ij}^- t_{ij},\ c_{ij0}^-\} \tag{4.7}$$

where c_{ij0}^- is a constant and the corresponding upper bound function is linear, $c_{ij}^+(t) = c_{ij}^+ - \alpha_{ij}^+ t_{ij}$, and verifies $c_{ij}^-(t) \le c_{ij}^+(t)\ \forall t \in T$. If this last condition was not verified it would be included in the set T, in fact, it would be equivalent to include a bound on the investment t_{ij}.

In order to linearize this new model, the following auxiliary variables are included

$$clow_{ij} = \max\{c_{ij}^- - \alpha_{ij}^- t_{ij},\ c_{ij0}^-\},\ \forall (i,j) \in A,$$

and

$$low_{ij} = \begin{cases} 1 & \text{if } c_{ij}^- - \alpha_{ij}^- t_{ij} \leq c_{ij0}^-, \\ 0 & \text{if } c_{ij}^- - \alpha_{ij}^- t_{ij} \geq c_{ij0}^-, \end{cases} \qquad \forall (i,j) \in A.$$

Using the variables low_{ij} we can rewrite $clow_{ij}$ as follows

$$clow_{ij} = low_{ij} c_{ij0}^- + (1 - low_{ij})(c_{ij}^- - \alpha_{ij}^- t_{ij}) \ \forall (i,j) \in A.$$

Finally, we define the set of constraints that guarantee that low_{ij} takes the correct value

$$low_{ij}(c_{ij0}^- - c_{ij}^- + \alpha_{ij}^- t_{ij}) + (1 - low_{ij})(c_{ij}^- - \alpha_{ij}^- t_{ij} - c_{ij0}^-) \geq 0 \ \forall (i,j) \in A.$$

Using this new set of variables and constraints, we can formulate the RSPPI with these piecewise linear lower bound cost functions as

$$\begin{aligned}
&\min \sum_{(i,j)\in A} (c_{ij}^+ - \alpha_{ij}^+ t_{ij}) x_{ij} - u_n \\
&\text{subject to} \\
&\quad - \sum_{(i,j)\in A} x_{ij} + \sum_{(j,k)\in A} x_{jk} = b_j, \quad j = 1, 2, ..., n, \\
&\quad u_j \leq u_i + clow_{ij} + (c_{ij}^+ - \alpha_{ij}^+ t_{ij} - clow_{ij}) x_{ij}, \quad \forall (i,j) \in A, \\
&\quad clow_{ij} = low_{ij} c_{ij0}^- + (1 - low_{ij})(c_{ij}^- - \alpha_{ij}^- t_{ij}), \ \forall (i,j) \in A, \\
&\quad low_{ij}(c_{ij0}^- - c_{ij}^- + \alpha_{ij}^- t_{ij}) + (1 - low_{ij})(c_{ij}^- - \alpha_{ij}^- t_{ij} - c_{ij0}^-) \geq 0, \ \forall (i,j) \in A, \\
&\quad c_{ij}^+ - \alpha_{ij}^+ t_{ij} \geq clow_{ij}, \quad \forall (i,j) \in A, \\
&\quad \sum_{(i,j)\in A} t_{ij} \leq 1, \\
&\quad u_1 = 0, \\
&\quad t_{ij} \geq 0, \quad \forall (i,j) \in A, \\
&\quad x_{ij}, low_{ij} \ \in \{0,1\}, \quad \forall (i,j) \in A,
\end{aligned}$$

where $b_1 = 1, \ b_n = -1, \ b_j = 0 \ \forall j \in V \setminus \{1, n\}$.

Observe that once again we obtain a quadratic objective function and quadratic constraints due to products $x_{ij}t_{ij}$, $clow_{ij}x_{ij}$ and $low_{ij}t_{ij}$. By defining a new set of variables as in (4.5) we can obtain a MILP Formulation using previous arguments.

The above procedure used to adapt the MILP formulation to this type of piecewise linear cost bound functions can also be generalized to manage general piecewise linear functions. Hence, it can also be used to have an *approximate* MILP formulation for those models using general cost bound functions after approximating them by piecewise linear functions.

Example *Robust SPP with Investments*

We will consider here an extension of the example proposed in Montemanni and Gambardella (2004) in which it will be allowed to make investments in order to improve the cost structure of the following network

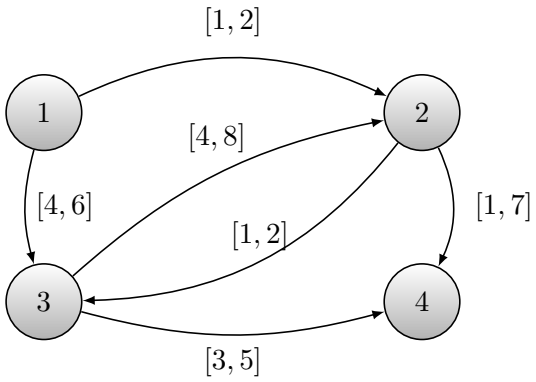

Figure 4.1: Network (V, A) for the Robust Shortest Path Problem.

The intervals depicted near the arcs represent the *static* uncertainty cost intervals. For this input, the robust shortest path optimal solution is $(1, 2), (2, 4)$ which reaches the minimum maximum regret of 3 units.

We will now consider that we can invest a total amount of one unit of a given resource and the cost bound functions are described as in (4.7) with $c_{ij0} = 0$ for all $(i, j) \in A$ and the following slopes

$$\alpha^{\pm}_{12} = 0,\ \alpha^{\pm}_{13} = 4,\ \alpha^{\pm}_{23} = 2,\ \alpha^{\pm}_{24} = 0.4,\ \alpha^{\pm}_{32} = 3.2,\ \alpha^{\pm}_{34} = 4$$

In the following Figure 4.2 it has been depicted the *dynamic* uncertainty cost intervals which depend on the investment carried out in each arc.

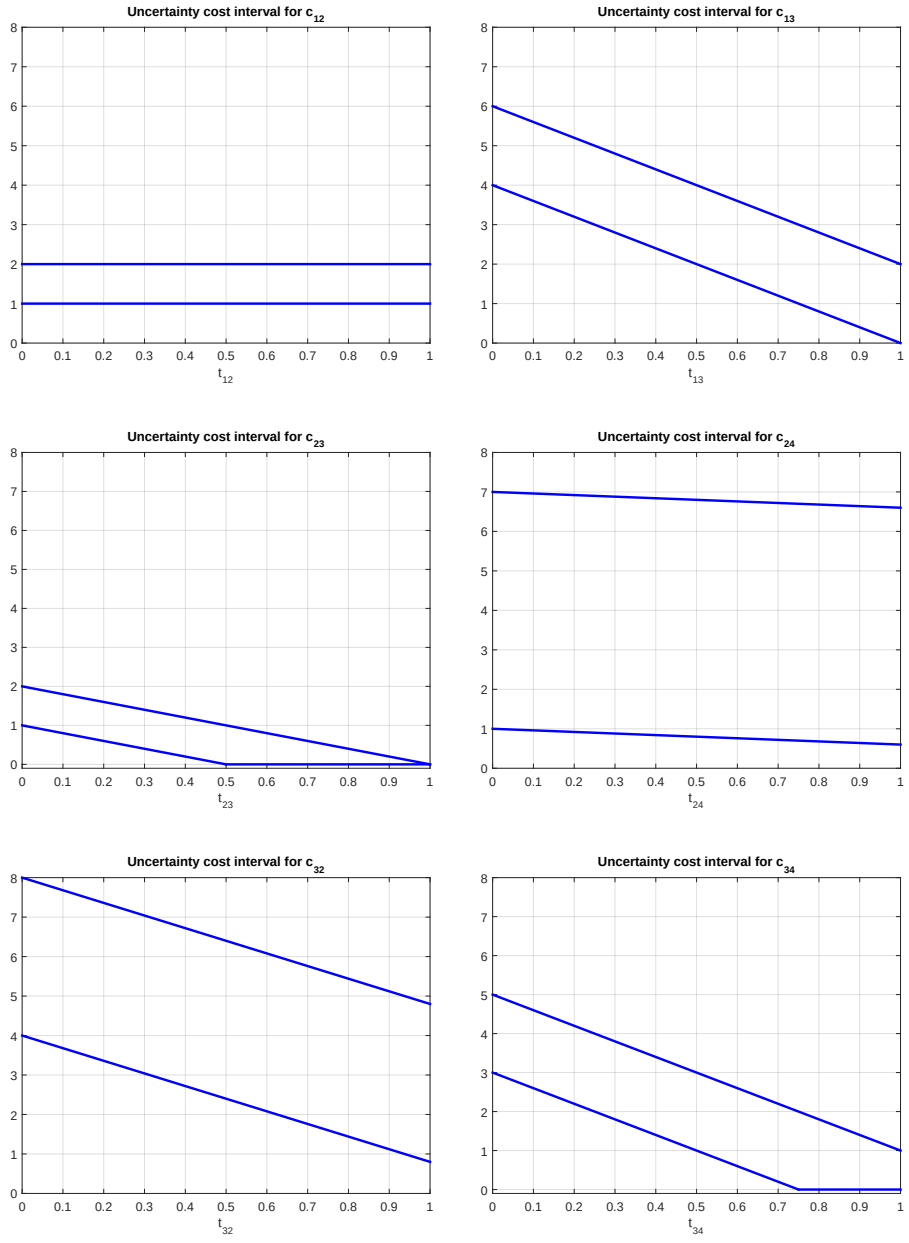

Figure 4.2: Dynamic uncertinty costs intervals depending on the investment.

Using solver Cplex, one can obtain the optimal solution

$$t_{34} = 1, \text{ and } x_{12} = 1,\ x_{23} = 1,\ x_{34} = 1$$

which illustrates how the cost structure of the network is improved by investing all the available resource in the arc $(3, 4)$, giving rise to the input network depicted in the figure 4.3 for the problem of finding the robust shortest path.

Now, the minimum maximum regret reached by the path $(1, 2), (2, 3), (3, 4)$ is 2.

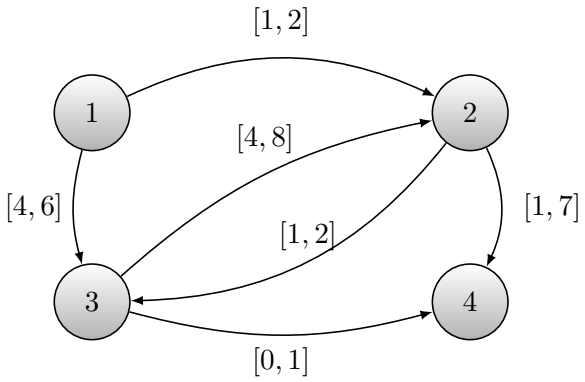

Figure 4.3: Network (V, A) for the Robust Shortest Path Problem with investment $t_{34} = 1$.

Hence, the behavior of the new solution with respect to the optimal path under the worst-case scenario has been improved after the investment made in the system. In other words, the consequence of the investment is the reduction of the risk that the decision maker must face up to the action of implementing a solution without the knowledge of the exact scenario that will occur.

RSPPI and other problems fitting into the formulation $(RCI - D)$ like assignment problems or general flow problems with investments are Strongly NP-hard, due to the fact that the minmax regret versions of these problems without investments, that is, when T is a singleton, are Strongly NP-hard (Section 1.1.1). This fact highlights the importance of having efficient methods that provide approximate solutions even when there exist mathematical formulations enabling us to solve the problem exactly.

4.4 Approximations

4.4.1 2-approximation result

Let us consider the problem (RCI) for a fixed $t \in T$,

$$\min_{x \in X} \left[R(x,t) := \max_{s \in S_t} \left\{ \sum_{(i,j) \in A} c_{ij}^s(t) x_{ij} - \min_{y \in X} \sum_{(i,j) \in A} c_{ij}^s(t) y_{ij} \right\} \right], \tag{4.8}$$

and let $\bar{c}(t)$ be the cost scenario where all its components are fixed as the mid-point of their corresponding uncertainty cost intervals defined by the investment $t \in T$ (mid-point scenario). One can have a 2-approximation (see Section 1.1.1, Proposition 4) of this problem by solving the following one

$$\begin{array}{ll} \min & \sum_{(i,j)\in A} \bar{c}_{ij}(t)x_{ij} \\ \text{subject to} & \\ & x \in X, \end{array} \qquad (\text{MRCPI}_t)$$

the deterministic problem associated to the mid-point scenario. Under the unimodularity condition on the set of feasible solutions X, imposed in Section 4.2, (MRCPI_t) is a Linear Programming problem that can be efficiently solved. Furthermore, there are sufficient optimality conditions that can be written in terms of basic feasible solutions allowing us making a parametric study with respect to the vector of investments.

Let us consider the function

$$t \in T \longrightarrow \bar{x}_t \in X,$$

assigning an arbitrary optimal solution $\bar{x}_t$ of the problem (MRCPI_t) to each vector of investments. Following Kasperski and Zielinski (2006), $\bar{x}_t$ is a 2-approximation of the problem (RCI) for a fixed $t \in T$, that is,

$$R(x_t^*, t) \leq R(\bar{x}_t, t) \leq 2R(x_t^*, t), \tag{4.9}$$

where $R(x,t)$ is the objective function minimized in (4.8) and x_t^* is one of its optimal solutions. It is easy to see that, by considering only a finite number of investments, one can have a 2-approximate solution for the problem (RCI) when X has the unimodularity property. In order to prove it, let us define the following finite family of sets

$$T_B = \left\{t \in T : \bar{c}_N(t) - \bar{c}_B(t)B^{-1}N \geq 0\right\}, \quad B \in \mathcal{B}, \tag{4.10}$$

where $\mathcal{B}$ is the set of all the basic feasible matrices B of $X = \{x : Mx = b, x \in [0,1]^{|A|}\}$. Here, we have used for simplicity, the same notation for X and for its continuous relaxation. For a given matrix B, N is the corresponding nonbasic matrix and $\bar{c}_B(t), \bar{c}_N(t)$ are the vectors of basic and nonbasic components of $\bar{c}(t)$. The family $\mathcal{B}$ allows to write the set of feasible investments as a finite union of T_B-sets

$$T = \bigcup_{B\in\mathcal{B}} T_B. \tag{4.11}$$

The previous equality holds because for any arbitrary investment $t \in T$, the problem (MRCPI_t) has, at least, a basic optimal solution (X is a bounded polyhedron) with an associated basic feasible matrix B for which $\bar{c}_N(t) - \bar{c}_B(t)B^{-1}N \geq 0$ is verified.

The set T_B represents the set of investments $t \in T$ for which the basic feasible solution associated with the matrix B is an optimal solution of the problem (MRCPI$_t$). Hence, this basic solution is a 2-approximation to the optimal solution of the problem (4.8) for every $t \in T_B$.

The finite union of T_B-sets (4.11) could eventually have only one nonempty set, let us say, $T_{B_1} \neq \emptyset$ and let x^1 be its corresponding basic feasible solution. Under this situation, any optimal solution (x^*, t^*) of the problem (RCI) must verify that $t^* \in T_{B_1}$, hence the equation (4.9) guarantees that

$$R(x^*, t^*) \leq R(x^1, t^*) \leq 2R(x^*, t^*),$$

that is, x^1 would be a 2-approximation for the problem (RCI). If this is not the case, but there is a relatively small set of basis defining a finite union of T_B-sets of T, one can still find efficiently a 2-approximation for the problem (RCI) by selecting for any one of the corresponding basic feasible solutions an investment minimizing its maximum regret. To see this, we will label as elements of a set K a minimal subfamily of basic feasible matrices B whose sets T_B define a finite union of T_B-sets of T,

$$T = \bigcup_{k \in K} T_{B_k}. \tag{4.12}$$

We have a set $X_0 = \{x^k, k \in K\}$, where x^k is the corresponding basic feasible solution for the matrix B_k, that is, x^k is optimal for the problem (P_t) for every $t \in T_{B_k}$. For each basic solution x^k we compute an investment reaching its minimum maximum regret $t(x^k)$ given by

$$\min_{t \in T} \max_{s \in S_t} R(x^k, t, s) =$$

$$= \min_{t \in T} \left\{ \sum_{(i,j) \in A} c_{ij}^+(t) x_{ij}^k - \min_{y \in X} \sum_{(i,j) \in A} \left[(c_{ij}^+(t) - c_{ij}^-(t)) x_{ij}^k + c_{ij}^-(t) \right] y_{ij} \right\}.$$

Note that, if $c_{ij}^+(t)$ and $c_{ij}^-(t)$ are linear functions, the problem of determining $t(x^i)$ is equivalent to solve a linear problem according to the formulation (4.4). If these cost bound functions are piecewise linear one can obtain an equivalent MILP by introducing binary variables as in the subsection 4.3.2.

Proposition 14. *The pair* $(\bar{x}, t(\bar{x}))$ *defined by*

$$(\bar{x}, t(\bar{x})) = argmin_{k \in K} R(x^k, t(x^k))$$

is an approximation of constant factor 2 for the minmax regret combinatorial optimization problem with investments (RCI).

Proof. If we denote by (x^*, t^*) an optimal solution of the problem (RCI):

$$R(x^*, t^*) \leq R(\bar{x}, t(\bar{x})).$$

As $T = \bigcup_{k \in K} T_{B_k}$, $\exists k_0 \in K$ such that $t^* \in T_{B_{k_0}}$, therefore:

$$R(x^*, t^*) \leq R(\bar{x}, t(\bar{x})) \leq R(x^{k_0}, t(x^{k_0})) \leq R(x^{k_0}, t^*) \leq 2R(x^*, t^*).$$

□

□

Unfortunately, obtaining a finite union of T_B-sets of T with a small set of basis $\{B^k : k \in K\}$ is, in general, a difficult task. Note that, if the cost bound functions $c_{ij}^{\pm}(t)$ are not linear, the set T_B do not need to be a convex set nor even a connected set. Hence, the development of a general method to find an efficient finite union of T_B-sets can not take advantage of good geometrical properties whilst generating all the feasible basis is not an option because it is not a polynomial process and it would be equivalent to solve the problem (RCI) by enumeration. In fact, if all the basic feasible matrices are generated the problem of obtaining an approximate solution is meaningless because, in this case, the pair $(\bar{x}, t(\bar{x}))$ defines by itself an optimal solution since X has the unimodularity property. In what follows, Proposition 14 will be used as the basis of a heuristic scheme to propose a solution *close* to a 2-approximate solution and an indicator of the quality of such a solution.

4.4.2 Heuristic

The algorithm proposed here generates a random sequence of investments in T, for each generated investment t, the problem (MRCPI$_t$) is solved and after that, the pair $(\bar{x}, t(\bar{x}))$ is obtained as in Proposition 14. If, in this sequence of investments, there is at least one of them in one of the T_B-sets containing the investment vector t^* of one of the optimal solution of (RCI) then, the sequence of inequalities of the proof of Proposition 14 can be applied and a 2-approximation solution is guaranteed. In order to ensure that this situation occurs with probability one, when the size of the random sample of investments increases to infinity (convergence *almost sure*), it is sufficient to check that a finite union of T_B-sets of T can always be selected in such a way that the probability of generating an investment of the random sample belonging to each T_B-set is strictly positive. If this is true and the sample is constituted of independent and identically distributed random vectors, the Strong Law of Large Numbers (SLLN) asserts this procedure will find a 2-approximation of the problem (RCI) with probability one when the size of the sample increases to infinity.

Before giving some conditions under which this argument can be applied, we will formally state the

Algorithm 6 Approximate algorithm

1: Choose a sample of investments $\mathbb{T} = \{t^1, ..., t^r\} \subset T$.
2: Solve the problem (P_t) for all $t \in \mathbb{T}$ in order to obtain the set $\mathbb{X}_0 = \{x^1, ..., x^r\}$, where x^i denotes an optimal solution of (P_{t^i}), $t^i \in \mathbb{T}$.
3: For each $x^i \in \mathbb{X}_0$ obtain its best investment $t(x^i)$ or one of them if there are more than one.
4: Choose as a *candidate* solution:

$$(\bar{x}, t(\bar{x})) \in argmin\{R(x^i, t(x^i)) : x^i \in \mathbb{X}_0\}. \tag{4.13}$$

Proposition 15. *If the sample of investments of the* Approximate algorithm *is independent and identically distributed according to an absolute distribution over T with strictly positive density and the cost bound functions $c_{ij}^{\pm}(t)$ are continuous in T for all $(i,j) \in A$, then the candidate solution* (4.13) *converges almost sure to a 2-approximation of the problem (RCI).*

Proof. Let us suppose that the set T has nonempty interior, we will show that there exists a finite union of T_B-sets of T like (4.11) with all the T_B-sets having nonempty interior. If T has an empty interior one can reason in the same way with the relative interior of T, that is, its interior within the affine hull of T. Once this is shown one has that there exists at least a T_B-set, with nonempty interior, containing a vector of optimal investments t^* for the problem (RCI). By the distributional imposed hypotheses, the probability of generating a vector of investments contained in this set is strictly positive, that is, the SLLN guarantees that almost sure one of this vectors will be generated as the size of the sample increases. Finally, we can use the argument stated in the first paragraph of this section to assert the almost sure convergence of the candidate solution (4.13) to a 2-approximation of the problem (RCI).

Let t be an interior point of T, for $\epsilon > 0$ sufficiently small, one can have $t \in B(t,\epsilon) \subset \text{int}(T)$, where $B(t,\epsilon)$ is a ball with its center at t and a radius $\epsilon > 0$ with respect to the Euclidean distance in $\Re^{|A|}$. Since this is a complete metric space, the union of every countable collection of closed sets with empty interior has empty interior (Baire space). Furthermore the functions $c_{ij}^{\pm}$ are all continuous, by its definition, (4.10), the sets T_B are closed. Hence the following set has an empty interior

$$\bigcup_{B' \in \mathcal{B}'} T_{B'}$$

where $\mathcal{B}'$ is the set of basic feasible matrices B' for which $\text{int}(T_{B'}) = \emptyset$. The fact

that this set has an empty interior means that

$$B(t,\epsilon) \nsubseteq \bigcup_{B' \in \mathcal{B}'} T_{B'}$$

for each $\epsilon > 0$, that is, since the number of feasible basis is finite and it is verified (4.11), one has that t is an accumulation (limit) point of one of the T_B-sets with $\text{int}(T_B) \neq \emptyset$. This set is closed so $t \in T_B$, that is,

$$\text{int}(T) \subseteq \bigcup_{B \in \mathcal{B} \setminus \mathcal{B}'} T_B.$$

Finally, as the finite union of closed sets is closed one has

$$T = \bigcup_{B \in \mathcal{B} \setminus \mathcal{B}'} T_B.$$

□ □

Remark 9. *This result remains valid for problems of type (RCI) in which one can have a 2-approximation for the problem* (4.8) *for a fixed* $t \in T$ *by solving the corresponding deterministic problem (MRCPI*$_t$*) under the mid-point scenario. Such problems do not neet to have as feasible set a polyhedron* X *with unimodular matrix (see for instance Conde (2010); Kasperski (2008); Kasperski and Zielinski (2006)).*

Although the SLLN ensures the convergence of this procedure to a 2-approximation if the above-mentioned conditions are verified, in practice, the size of the sequence of investments that should be generated is unknown. Hence, it is important to have an indicator of the quality of the proposed solution when the process is stopped and no way of guaranteeing a 2-approximation is known.

Hereafter, some results of Conde (2010) are adapted to our optimization model in order to bound the error of the solution proposed by the heuristic algorithm. Let us define

$$\delta^*(x,t) = \max_{s \in S_t} \left\{ 0, \sum_{(i,j) \in A} c_{ij}^s(t) x_{ij} \right\}.$$

It is easy to see that we can rewrite the maximum regret of a given solution x under an investment t as

$$R(x,t) = \max_{y \in X} \delta^*(x-y,t). \tag{4.14}$$

Therefore the problem (RCI) can be expressed as

$$\min_{t \in T} \min_{x \in X} \max_{y \in X} \delta^*(x-y,t).$$

The following properties can be used in order to obtain a bound on $R(\bar{x}, t(\bar{x}))$.

Proposition 16. *Given $x \in X$, $t \in T$, we have that*

$$R(x,t) \geq \sum_{(i,j)\in A} c_{ij}^{+}(t)\max\{x_{ij} - y_{ij}, 0\} - \sum_{(i,j)\in A} c_{ij}^{-}(t)\max\{y_{ij} - x_{ij}, 0\}, \ \forall y \in X.$$

Proof. By (4.14) we know that

$$R(x,t) \geq \delta^*(x-y,t) \geq \sum_{(i,j)\in A} c_{ij}^{*}(x_{ij} - y_{ij}), \quad \forall y \in X,$$

where

$$c^* = \begin{cases} c_{ij}^{*} = c_{ij}^{+}(t) & if\ x_{ij} \geq y_{ij}, \\ c_{ij}^{*} = c_{ij}^{-}(t) & if\ x_{ij} < y_{ij}. \end{cases} \qquad (i,j) \in A.$$

Then, we obtain the inequality

$$R(x,t) \geq \sum_{(i,j)\in A} c_{ij}^{+}(t)\max\{x_{ij} - y_{ij}, 0\} - \sum_{(i,j)\in A} c_{ij}^{-}(t)\max\{y_{ij} - x_{ij}, 0\}.$$

□ □

Proposition 17. *Given $x, y \in X$, $t \in T$, we have:*

$$R(x,t) \leq R(y,t) + \delta^*(x-y,t).$$

Proof. Given $x, y \in X$, $t \in T$, by the triangular property of the function $\delta^*(.)$ we have

$$\delta^*(x-v,t) \leq \delta^*(x-y,t) + \delta^*(y-v,t), \ \ \forall v \in X.$$

Now, by taking the maximum on $v \in X$ in the previous inequality, one has

$$R(x,t) \leq \delta^*(x-y,t) + R(y,t).$$

□ □

The following auxiliary optimization problem will give us an indicator of the quality of the approximation when the algorithm is stopped. Its optimum is an upper bound on the difference between the objective value of the candidate solution and twice the optimum of the problem (RCI). The formulation of this auxiliary

problem is as follows

$$\begin{array}{rll} b = & \max \min_{\hat{x} \in \mathbb{X}_0} & 2 \sum_{(i,j) \in A} \bar{c}_{ij}(t)(\hat{x}_{ij} - x_{ij}) \\ & \text{subject to} & \\ & & t \in T, \\ & & x \in X, \end{array} \tag{P_b}$$

where the constant b is always finite since T and X are compact sets. Observe that, in the case in which there exists a common optimal solution $\hat{x} \in \mathbb{X}_0$ of (P_t) for every $t \in T$, the constant b is zero. This also happens when there is at least an investment vector of the sample $\mathbb{T}$ in each element of a finite union of T_B-sets of T.

It is important to note that the problem (P_b) only needs to be solved at the end of the approximate algorithm if one wants to have a bound on the incurred error. In the numerical experiments presented in the next section it will be solved after every generated sample of investments is processed in order to show an indicator of the quality of the approximation. However, if this bound is not necessary, as in the case in which the approximation is used as a starting solution for an exact algorithm, the problem (P_b) does not need to be solved. In any case, as it is justified later, from a computational viewpoint, it is much more efficient to solve (P_b) than (RCI).

After having solved the problem (P_b) one has the following bound on the error of the approximate solution

Proposition 18. *The maximum regret of a solution $(\bar{x}, t(\bar{x}))$ defined as in (4.13) verifies that*

$$R(\bar{x}, t(\bar{x})) \leq b + 2R^*,$$

where R^* denotes the optimal value of the problem (RCI), and b is the optimal objective value of the problem (P_b).

Proof. Let (x^*, t^*) be an optimal solution of the problem (RCI), then there exists $\hat{x} \in \mathbb{X}_0$ such that

$$2 \sum_{(i,j) \in A} \bar{c}_{ij}(t^*)(\hat{x}_{ij} - x^*_{ij}) = \sum_{(i,j) \in A} (c^+_{ij}(t^*) + c^-_{ij}(t^*))(\hat{x}_{ij} - x^*_{ij}) \leq b. \tag{4.15}$$

Let us suppose that $R(x^*, t^*) < R(\bar{x}, t(\bar{x}))$ then, by Proposition 17 one has that

$$\begin{gathered} 0 < R(\bar{x}, t(\bar{x})) - R(x^*, t^*) \leq R(\hat{x}, t(\hat{x})) - R(x^*, t^*) \leq \\ \leq R(\hat{x}, t^*) - R(x^*, t^*) \leq \delta^*(\hat{x} - x^*, t^*) = \\ = \textstyle\sum_{(i,j) \in A} c^+_{ij}(t^*) \max\{\hat{x}_{ij} - x^*_{ij}, 0\} - \sum c^-_{ij}(t^*), \max\{x^*_{ij} - \hat{x}_{ij}, 0\} \end{gathered} \tag{4.16}$$

where the first inequality hold by definition of $\bar{x}$ and $t(\hat{x})$.

We can now write (4.15) as

$$\begin{aligned}&\textstyle\sum_{(i,j)\in A}(c^+_{ij}(t^*)+c^-_{ij}(t^*))\max\{\hat{x}_{ij}-x^*_{ij},0\}-\\ -&\textstyle\sum_{(i,j)\in A}(c^+_{ij}(t^*)+c^-_{ij}(t^*))\max\{x^*_{ij}-\hat{x}_{ij},0\}\leq b.\end{aligned}$$

In particular,

$$\sum_{(i,j)\in A} c^+_{ij}(t^*)\max\{\hat{x}_{ij}-x^*_{ij},0\}-\sum_{(i,j)\in A} c^-_{ij}(t^*)\max\{x^*_{ij}-\hat{x}_{ij},0\}\leq$$

$$\leq b+\sum_{(i,j)\in A} c^+_{ij}(t^*)\max\{x^*_{ij}-\hat{x}_{ij},0\}-\sum_{(i,j)\in A} c^-_{ij}(t^*)\max\{\hat{x}_{ij}-x^*_{ij},0\}\leq$$

$$\leq b+R(x^*,t^*).$$

Therefore, using (4.16), we finally obtain the bound

$$R(x^*,t^*)\leq R(\bar{x},t(\bar{x}))\leq b+2R(x^*,t^*).$$

□ □

In order to solve the problem (P_b) we can define a new variable γ to move the objective function to the constraints, that is,

$$\begin{array}{lll} b= & \max & \gamma \\ & \text{subject to} & \\ & & t\in T, \\ & & x\in X, \\ & & \gamma\leq\langle 2\bar{c}(t),\hat{x}-x\rangle,\forall\hat{x}\in\mathbb{X}_0. \end{array} \qquad (P'_b)$$

In the case of functions $c^{\pm}_{ij}(t)$ with linear or piecewise linear structure, this last problem can be linearized by using auxiliary sets of variables, as in (4.5), to obtain a MILP problem.

Observe that problem (P'_b) has the same number of integer decision variables than the exact formulation, nevertheless the number of quadratic constraints to be linearized is substantially reduced from three per each arc in the exact formulation to, at most, one per each investment of the sample $\mathbb{T}$ (exactly to $|\mathbb{X}_0|$ quadratic constraints). Despite the fact that problem (P'_b) can have a maximum of $|\mathbb{T}|$ quadratic constraints, our numerical experiments show us that this number is, in practice, smaller due to the fact that different investments considered in the sample can have the same optimal path. In the next section we will show some numerical experiments in which this significant reduction on the number of quadratic constraints will result

in a considerable decrease of the computational times spent to solve (P'_b) with respect to the exact formulation, in particular, for big graphs.

To end this section we will apply the above results to the example of section 4.3.2

Example *Robust SPP with Investments* (continuation)

1. We consider 3 different feasible investments:

$$\mathbb{T} = \{t^1 = (0.125, 0.125, 0.125, 0.125, 0.125, 0.125),\ t^2 = (0, 0.5, 0, 0, 0, 0.5),$$

$$t^3 = (0, 0.25, 0, 0.25, 0.25, 0.25)\},$$

   where $t^i = (t^i_{12}, t^i_{13}, t^i_{23}, t^i_{24}, t^i_{32}, t^i_{34}),\ i = 1, 2, 3.$

2. We solve $P_{t^1},\ P_{t^2},\ P_{t^3}$ obtaining the following set $\mathbb{X}_0$:

$$\mathbb{X}_0 = \{x^1 = (1, 0, 0, 1, 0, 0),\ x^2 = (1, 0, 1, 0, 0, 1),\ x^3 = (1, 0, 0, 1, 0, 0)\},$$

   where $x^i = (x^i_{12}, x^i_{13}, x^i_{23}, x^i_{24}, x^i_{32}, x^i_{34}),\ i = 1, 2, 3.$

3. We compute the best investment,or one of them if there are more than one, for each path x^1, x^2 (observe that $x^1 = x^3$):

$$t(x^1) = (0, 0, 0, 1, 0, 0),\ t(x^2) = (0, 0, 0, 0, 0, 1),$$

   so, for each pair, we obtain the following values:

$$R(x^1, t(x^1)) = 2.6,\ R(x^2, t(x^2)) = 2.$$

4. Then the solution candidate proposed by the heuristic would be

$$x^2 = (1, 0, 1, 0, 0, 1),\ t(x^2) = (0, 0, 0, 0, 0, 1),\ R(x^2, t(x^2)) = 2$$

   which coincides with the optimal solution of the problem.

   Finally, by using the result given in Proposition 18, we can bound the error of this approximate solution after solving the problem (P'_b). In this case, $b = 1$, so we have

$$R(x^*, t^*) \leq R(x^2, t(x^2)) \leq 1 + 2R(x^*, t^*).$$

4.5 Computational experiment

In this last section we will show a numerical experiment conducted to check the effectiveness of the approximate algorithm of the previous section. The experiment concerns the minmax regret shortest path problem with piecewise linear cost bound functions (section 4.3.2) and feasible set of investments given by

$$T = \left\{ (t_{ij})_{(i,j)\in A} : \sum_{(i,j)\in A} t_{ij} \leq 1,\, c_{ij}^{-}(t) \leq c_{ij}^{+}(t),\, t_{ij} \geq 0, \forall (i,j) \in A \right\}.$$

Instances of this problem were randomly generated and solved both exactly, by using the formulation of Section 4.3.2, and with the approximation procedure of the previous section.

For a given number of nodes n, the arc (i,j), $i = 1,\ldots,n$ and $j = i+1,\ldots,\min\{i+s,n\}$, is independently created with a fixed probability p, where s is also a fixed parameter for each sample of instances. The networks without at least a path between the node 1 and n are discarded. This type of graphs allows us to limit the number of neighbors of the nodes, that is, those nodes that can be accessed directly from the considered one, providing feasible paths passing through a significant number of nodes like it is frequent in real applications.

The constants c_{ij0}^{-}, c_{ij}^{-} and c_{ij}^{+} of the piecewise linear cost bound functions were generated randomly in such a way that $c_{ij0}^{-}, c_{ij}^{-} \in [0,10]$, $c_{ij0}^{-} \leq c_{ij}^{-}$ and $c_{ij}^{+} \in [c_{ij}^{-}, c_{ij}^{-}+10]$. The slopes $\alpha_{ij}^{\pm}$ of these functions are random values uniformly distributed in the interval $[0,\alpha]$ where α is the 0.1% of the sum of the values c_{ij0}^{-} (*type I*) or the 1% of this sum (*type II*). We consider these two cases in order to check the effects of different scales for the slopes of the cost bound functions in the performance of the approximate algorithm. Note that the slopes generated in our experiment allow cost variations of even the 100α% of the reference cost incurred if no investment is in the arc since the components of the investment vector have been normalized according to (4.3).

The number of feasible investments "*nt*" used by the approximate algorithm, that is, the cardinality of the set $\mathbb{T}$, was taken as 15 for all instances. This number was chosen after doing some experiments with $nt = 15, 25, 50$ and observing that the increase of this number increased also the computational time of the heuristic but the bound b remained almost the same. These investments for each instance of the problem were also randomly generated according to the following procedure:

- A subset of arcs was chosen randomly using independent Bernoulli distributions with probability of success of 2% of the sum of the reference values c_{ij0}^{-} divided by the number of arcs, that is, this probability is always lower than 0.2 since

the values c^{-}_{ij0} belong to $[0, 10]$.

- If the above subset has h arcs, a random vector is generated according to a uniform distribution in $[0, 1]^{h+1}$. This vector was normalized to one and h of its components were assigned as the investments in the h arcs included in the subset; the remaining component is considered the slack with respect to the total amount of available resources (one unit).

- The investments in the arcs not included in the subset were fixed to zero.

Note that this procedure verify the conditions of Proposition 15 since the density of the investment vector is a mixture of positive densities in subsets covering the set T.

The computational experiment was carried out on a personal computer with Intel ® Core (TM) i7-4720HQ, 2.60GHz with 16384 MB RAM. The optimization problems were solved exactly by using CPLEX Version: 12.6.1.0 and the approximate algorithm was written as an IBM ILOG CPLEX Optimization Studio project, Version: 12.6.1.0.

In tables 1 to 4 it is shown the results obtained for 280 different instances of the problem varying the number of nodes of each network and taking the probability of generating an arc as $p = 0.6$. The averages of the number of generated arcs appear in the column (m). In each table the parameter s is fixed, $s \in \{3, 10\}$ (see the heading of each table) and the parameter n takes the values $150, 300, 450, 600, 750, 900$ and 1050.

Once α, s and n were fixed, each row of the table was completed with the averages of the computational times of the exact algorithm (CPU^1), the heuristic algorithm (CPU^2) and the resolution of the problem (P'_b) (CPU^3) for 10 different instances of the problem. It is also shown in each row the average values of the fraction (r) of the objective value of the approximate solution (numerator) over the optimal objective value of the problem (RCI) (denominator) and the fraction (b') of the bound b (numerator) over the optimal value of the problem (RCI) (denominator).

s=3	α(type I)	**EXACT FORM.**	**HEURISTIC**		**(Pb')**	
n	**m**	**CPU^1**	**CPU^2**	**r**	**CPU^3**	**b'**
150	267.4	1.0	3.1	1.012	1.0	0.004
300	561.8	3.5	4.0	1.027	3.0	0.006
450	832.1	11.5	5.6	1.029	6.9	0.006
600	1100.7	49.5	7.5	1.028	13.6	0.007
750	1376.9	36.7	7.7	1.021	19.2	0.007
900	1663.7	167.9	9.6	1.023	28.7	0.007
1050	19506.0	**	11.2	1.025*	39.2	0.007*

Table 4.1: Computational Results for $s = 3$, α(type I).

s=3	α(type II)	**EXACT FORM.**	**HEURISTIC**		**(Pb')**	
n	**m**	**CPU**[1]	**CPU**[2]	**r**	**CPU**[3]	**b'**
150	282.4	1.1	3.7	1.029	1.0	0.053
300	547.7	3.5	5.3	1.023	4.0	0.055
450	824.2	9.2	7.1	1.026	8.9	0.068
600	1108.4	24.1	9.2	1.024	17.9	0.064
750	1380.1	24.6	9.6	1.029	24.3	0.066
900	1678.6	65.3	10.7	1.026	34.0	0.067
1050	1969.0	**	12.8	1.018 *	50.6	0.062 *

Table 4.2: Computational Results for $s = 3$, α(type II).
By * we mean that the corresponding value was obtained considering just the problems solved exactly, fewer than 10 instances. By ** we mean that the exact algorithm was stopped after 30 minutes of CPU without finding an exact solution.

From Proposition 18 one has the theoretical bound on the optimal objective value of the problem (RCI)

$$R(\bar{x}, t(\bar{x})) \leq b + 2R^*,$$

which guarantees us an upper bound on the indicator r greater or equal to 2. However, experimentally we observe that this bound is very conservative because most of the estimations of the parameter r using the corresponding sample average are lower than 1.06, this is, in most cases the candidate solution is better than a 1.06-approximation, which is a solution very close to an optimal one. In the worst observed cases, the candidate solution was better than a 1.12-approximation. Moreover, these approximations were obtained, on average, in a fraction of 0.21 of the time used by the exact formulation.

s=10	α(type I)	**EXACT FORM.**	**HEURISTIC**		**(Pb')**	
n	**m**	**CPU**[1]	**CPU**[2]	**r**	**CPU**[3]	**b'**
150	826.0	2.2	5.0	1.055	3.0	0.032
300	1726.6	12.0	9.1	1.046	7.7	0.052
450	2635.9	41.8	14.2	1.062	15.1	0.060
600	3522.4	118.4	23.1	1.055	33.2	0.055
750	4495.8	166.4	26.8	1.051	45.6	0.065
900	5371.0	303.7	32.5	1.058	78.2	0.059
1050	6299.1	**	41.6	1.055*	142.9	0.065*

Table 4.3: Computational Results for $s = 10$, α(type I).
By * we mean that the corresponding value was obtained considering just the problems solved exactly, fewer than 10 instances. By ** we mean that the exact algorithm was stopped after 30 minutes of CPU without finding an exact solution.

If we compare the computational times required to solve the exact formulation

s=10	α(type II)	**EXACT FORM.**	**HEURISTIC**		**(Pb')**	
n	**m**	**CPU**[1]	**CPU**[2]	**r**	**CPU**[3]	**b'**
150	856.8	2.9	5.4	1.049	3.2	0.758
300	1753.1	20.0	10.4	1.099	11.8	0.798
450	2667.8	43.6	15.9	1.103	23.7	0.809
600	3571.4	99.1	22.9	1.122	50.9	0.884
750	4499.8	168.7	28.8	1.091	72.8	0.857
900	5412.4	274.1	36.4	1.117	104.1	0.868
1050	6294.2	328.2	44.9	1.098	144	0.855

Table 4.4: Computational Results for $s = 10$, α(type II).
By * we mean that the corresponding value was obtained considering just the problems solved exactly, fewer than 10 instances. By ** we mean that the exact algorithm was stopped after 30 minutes of CPU without finding an exact solution.

and the ones required to obtain approximate solutions we observe that, when the number of nodes is greater than 300, the heuristic is always faster than the exact formulation. The difference between these two computational time averages increases substantially with the rise in the number of nodes in the graph. This good behavior of the heuristic procedure, in terms of its computational times, compared with the exact algorithm allows us to find approximate solutions for large instances of the problem for which the exact formulation can not be solved within the time limit considered (30 mins of CPU). As it can be seen in the three first tables, for $n = 1050$ there are instances whose MILP formulations could not be solved within the limit of 30 minutes of CPU. However, the heuristic spent less than 45 seconds to find an approximate solution in some of these instances. In fact, although it has not been included in the tables, the heuristic found approximate solutions for instances of the problem with 6000 nodes and more than 13200 arcs in less than 4 minutes. This is very interesting since the approximate solution can also be used as a starting solution for exact methods with the consequent saving in computational time due to the fact that a wide subset of feasible solutions can be cut by bounding the objective function.

Finally, we comment the experimental results obtained about the bound b on the optimal value of the problem (P_b). We can see that the estimations of this indicator are relatively small compared with the optimum of the problem in most instances. However, in the last table one can notice a significative increase of these estimations. The reason of this fact could be the greater arc density of the generated networks ($s = 10$) together with the high level of variation allowed in the coefficients c_{ij} (α type II). The enlargement in the ranges of variation of the cost coefficients defining the feasible scenarios would produce, at least as a trend, the increase in the number of the elements of the finite union of T_B-sets (4.11) needed to cover the feasible

set of investments. As it was said in the previous section, in order to obtain a 2-approximation it is needed to have at least a vector of investments of the set $\mathbb{T}$ in the interior of each T_B-set with non-empty interior. Hence, the greater number of T_B-sets, the worse the bound b (as a trend), for a fixed number of generated investments in the set T.

4.6 Conclusions

In this chapter we have proposed an optimization model under uncertainty in the cost coefficients of a linear objective function. Every configuration of possible costs is called a scenario and it is assumed this vector belongs to a hypercube S_t which depends on a set of investments $t \in T$ carried out in the system. It was adopted as optimization criterion the minimization of the maximum regret over the set of possible scenarios and possible investments. Hence, this model seeks a set of investments under which there exists at least a solution with a performance *close* to the optimal one under each possible scenario, that is, the objective of the investments is to reduce the uncertainty about the performance of the implemented solution. Some rationality criteria had been considered in order to model the modifications undergone by the set of possible scenarios under every feasible investment.

We have obtained exact formulations for different choices of the structure of the set of scenarios S_t that allow us to solve the problem using large-scale optimization software packages. However, according to our numerical experiments, the computational cost increases quickly with the size of the problem. This justifies the study of approximate methods that could obtain a good starting solution in order to speed up the execution of a subsequent exact algorithm by eliminating a wide subset of feasible solutions. The proposed approximation is based on the resolution of a sequence of deterministic problems with linear structure, thus efficiently solvable by standard optimization algorithms. In addition, it is possible to find a bound on the incurred error to assess the quality of the approximate solution. This bound extends previous results on constant factor approximations for minmax regret optimization problems where no investment is allowed. The numerical comparison of the approximate algorithm with respect to exact methods points out that these approximations could be efficiently applied in the design of specialized algorithms to solve new robust combinatorial optimization problems. The analysis of how any piece of relevant information can be extracted from the data and from the structure of the problem in order to guide the generation of the sample of investments used in the approximate algorithm could give rise to interesting results. This research line will be explored in the short future.

Chapter 5

Bilevel Portfolio Selection Problem with Pricing decisions on transaction costs

In this chapter, we present new price setting models in the Portfolio Optimization area. Specifically, we assume in our models, as a novelty, that the unit transaction costs, charged for investing in some assets of the portfolio, must be decided in the decision process by the financial intermediary, instead of assuming them given. Henceforth, there are two different decision-makers involved in the Portfolio Problem: the financial intermediary, setting the transaction costs, and the investor, selecting a portfolio. The transaction costs are set trying to maximize the bank's revenue, and the portfolio is chosen trying to minimize the risk and ensuring an expected profit. We assume a hierarchical decision order, and therefore, we present different leader-follower versions of the model: financial intermediary-leader, investor-leader, and social welfare models; and we analyze their properties. Moreover, we develop Mixed Integer Linear Programming formulations for some of the models and effective algorithms for some others. Finally, we report on some computational experiments performed on real data taken from the IBEX 35, the main benchmark stock exchange index of the Spanish stock market, and analyze and compare the results obtained by the different models.

5.1 Introduction

The classical model in portfolio optimization was originally proposed by Markowitz (1952). This model has served as the initial point for the development of modern portfolio financial theory. Over time, portfolio optimization problems have become more realistic, incorporating real-life aspects that make the resulting portfolios more cost effective than the alternatives that do not consider them (Castro et al. (2011); Kolm et al. (2014); Mansini et al. (2014, 2015)). Transaction costs can be seen as one of these important actual features to be included in portfolio optimization. These costs are those incurred by the investors when buying and selling assets on real financial markets, charged by the brokers or the financial institutions playing the role of intermediary. Transaction costs usually include banks and brokers comissions, fees, rates, etc. These commissions/fees/rates have a direct impact on the portfolio, specially for individual or small investors, since they will determine the net returns, reducing them and decreasing also the budget available for future investments (Baule (2010); Baumann and Trautmann (2013)).

To the best of our knowledge, in the existing literature, transaction costs are assumed to be given (Mansini et al. (2014, 2015)). They can be a fixed cost applied to each selected security in the portfolio (see e.g. Baule (2010); Baumann and Trautmann (2013); Kellerer et al. (2000); Mansini et al. (2014, 2015); Valle et al. (2014); Woodside-Oriakhi et al. (2013) and the references therein); or a variable rate to be paid which depends on the amount invested on each security included in the port-

folio. This dependence can be proportional or be given by a fixed cost that is only charged if the amount invested exceeds a given threshold, or some other functional form (see e.g. Baule (2010); Konno et al. (2005); Le Thi et al. (2009); Mansini et al. (2014, 2015) and the references therein). But in any case, unit transaction costs are known and predetermined in the optimization process. Nevertheless, it is meaningful to analyze the situations where transaction costs can be decision variables on financial institutions' hands so that they are set trying to maximize its own profit as part of the decision process that leads to optimal portfolios for investors.

The portfolio optimization problem considered in this chapter is based on a single-period model of investment and incorporates a pricing aspect on the transaction costs. We assume that there are two decision-makers involved in the situation: the investor and the financial institution (that we will call from now on "the bank" for simplicity). At the beginning of a period, an investor allocates the capital among various assets and during the investment period, each asset generates a random rate of return. Moreover, we consider that the bank can charge some transaction costs on the securities selected by the investor trying to maximize its benefits. This is a pricing phase in which the bank makes the decision on how much is going to charge to the traded securities. Considering transaction costs as a decision variable of the model is a novel element in portfolio optimization and it is one of the main contributions of this chapter. Then, at the end of the period, the result for the investor is a change of the capital invested (increased or decreased) which is measured by the weighted average of the individual rates of return minus transactions costs. On the other hand, the result for the bank is the amount paid by the investor which depends on the prices set on the traded securities and the portfolio selected by the investor.

Based on the structure of financial markets, we assume a hierarchical relationship between the parties involved in the portfolio problem, that is, we consider a natural model in which the bank sets the prices first, trying to anticipate the rational response of the investor. This hierarchical analysis of the portfolio problem has not been addressed before and it is another contribution of our chapter. Once the prices are fixed, the investor chooses her optimal portfolio. For the sake of completeness, we also analyze the case in which the investor chooses her portfolio first, and after that, the bank sets the transaction costs. In order to model these hierarchical structures, we use a bilevel optimization approach (see Section 1.1.2). Furthermore, we consider a social welfare model, that is, a model in which both, bank and investor, cooperate to maximize their returns. We assume in the different models that all economic or financial information is common knowledge and that all the decision-makers in the problem have access to it.

The contributions of this chapter can be summarized in the following: 1. it incorporates for the first time, the above hierarchical approach with two-levels of

decision-makers on portfolio optimization problems (the bank sets transaction costs trying to maximize its benefits, whereas the investor minimizes risk while ensuring a given expected return (Benati (2003, 2015))); 2. it introduces transaction costs as decision variables on financial institutions' hands; 3. it develops different bilevel programming formulations to obtain optimal solutions for the considered models.

The rest of the chapter is organized as follows. Section 5.2 states the preliminaries and the notation used throughout the chapter. In Section 5.3 we present the models description, the formulations and the resolution algorithms. Particularly, in Section 5.3.1, we present the model in which the bank is the leader and we develop two different MILP formulations to solve such problem; in Section 5.3.2, we introduce the investor-leader model and develop a Linear Programming (LP) formulation for it. In the more general case where additional constraints are required on the portfolio selection, we present a convergent iterative algorithm based on an "add hoc" decomposition of the model. Next, in Section 5.3.3, it is addressed a social welfare model. There, we propose a MILP formulation and an algorithm based on Benders decomposition for solving the problem. In Section 5.4, we report on the computational study of the different models discussed in the previous sections. Our results are based on real data taken from IBEX 35, the main benchmark stock exchange index of the Spanish stock market. Finally, Section 5.5 concludes the chapter.

5.2 Preliminaries

Let $N = \{1, ..., n\}$ be the set of securities considered for an investment, and $B \subseteq N$ a subset of securities in which the bank can charge transaction costs to the investor. In most cases, $B = N$, but there is no loss of generality to consider that B is a proper subset of N.

On the one hand, we assume that the bank can price security $j \in B$ from a discrete set, with cardinality s_j, of admissible prices, $\mathbb{P}_j = \{c_{j1}, ..., c_{js_j}\}$, and the bank's goal is to maximize its benefit. Further, we consider the case in which the price charged by the bank per security is proportional to the amount invested in such security. In other words, the bank's decision variables are unit transaction costs (commissions, fees, rates, ...) to be charged to the securities.

Let $x = (x_j)_{j=1,...,n}$ denote a vector of decision variables weighting x_j the proportion of the total amount invested on security j in the portfolio. We only suppose that the invested capital can not exceed the available budget, i.e.

$$x : \sum_{j=1}^{n} x_j \leq 1, \; x_j \geq 0, \qquad \text{for } j = 1, ..., n.$$

This budget constraint is the minimum requirement on the structure of the portfolios.

Nevertheless, without loss of generality, we could have assumed that some other linear constraints are imposed on the structure of the requested portfolio x. All the results in this chapter can be easily extended to more general situations that consider polyhedral sets of constraints defining the admissible set of portfolios.

Let us denote by p_j the value chosen by the bank to price security j. Then, for a given portfolio x (fixed), the problem faced by the bank can be modeled using the following set of binary decision variables: $a_{jk} = 1$ if price c_{jk} is assigned to p_j, this is, if $p_j = c_{jk}$ and $a_{jk} = 0$ otherwise. Thus, to maximize its profit the bank solves the following problem:

$$\max \sum_{j \in B} p_j x_j \qquad \textbf{(PricP)}$$

$$\text{s.t. } p_j = \sum_{k=1}^{s_j} c_{jk} a_{jk}, \quad j \in B, \tag{5.1}$$

$$\sum_{k=1}^{s_j} a_{jk} = 1, \qquad j \in B, \tag{5.2}$$

$$a_{jk} \in \{0,1\}, \qquad j \in B, k = 1, ..., s_j. \tag{5.3}$$

If no further constraint is imposed on prices the above is a valid formulation. However, in general, we will assume without loss of generality that the set of prices for the bank can be restricted to belong to some polyhedron $\mathbb{P}$, allowing $\mathbb{P} = \mathbb{R}_+^{|B|}$. This can be easily included in the above formulation with the following constraint:

$$p \in \mathbb{P}. \tag{5.4}$$

We observe that, if x is known, and constraint (5.4) is not included, the above problem is easy to solve (see Proposition 21): the bank will set prices to the maximum ones among those available for each security. Nevertheless, if the portfolio is unknown (to be decided by the investor) or a more general polyhedron is considered, the problem becomes more difficult.

On the other hand, we suppose that the investor wants to reduce the risk of its investment while ensuring a given expected return. At this point, several risk measures could be considered, among them variance of returns, Mean Absolute Deviation (MAD), Conditional Value at Risk (CVaR), Gini's Mean Difference, etcetera, (here, we refer the reader to Mansini et al. (2003) for further details on the topic). In this chapter, we have focused on a portfolio optimization problem based on the CVaR measure. This risk measure aims to avoid large losses: for a specific probability level α, the CVaR measures the conditional expectation of the smallest returns (largest

losses) with a cumulative probability α, that is, the average return of the given size (quantile) of worst realizations Mansini et al. (2003); Puerto et al. (2017); Rockafellar et al. (2000). Therefore, we assume that the investor's goals are to maximize its CVaR and, at the same time, to ensure that a minimum expected reward μ_0 is obtained with her portfolio.

In order to model the above situation, we consider that the rate of return of each security $j \in N$ is represented by a random variable R_j with a given mean $\mu_j = E(R_j)$. Each portfolio x defines a random variable $R_x = \sum_{j=1}^{n} R_j x_j$ that represents the portfolio rate of return (its expected value can be computed as $\mu(x) = \sum_{j=1}^{n} \mu_j x_j$). We consider T scenarios, each of them with probability π_t, $t = 1, ..., T$, and assume that for each random variable R_j its realization, r_{jt}, under the scenario t is known. Thus, once the bank has set its prices, p, the realization of the portfolio rates of return R_x under scenario t is given as $y_t = \sum_{j=1}^{n} r_{jt} x_j - \sum_{i \in B} p_i x_i$.

With this information, we assume that our investor wants to maximize the CVaR_α, namely the conditional expectation of the smallest returns with cumulated probability α, while ensuring a minimum expected return μ_0. Thus, the portfolio optimization model that the investor wants to solve can be formulated as:

$$\max \eta - \frac{1}{\alpha} \sum_{t=1}^{T} \pi_t d_t \qquad \textbf{(CVaRP)}$$

$$\text{s.t. } d_t \geq \eta - y_t, \qquad t = 1, ..., T, \tag{5.5}$$

$$y_t = \sum_{j=1}^{n} r_{jt} x_j - \sum_{i \in B} p_i x_i, \quad t = 1, ..., T, \tag{5.6}$$

$$\sum_{j=1}^{n} x_j \leq 1, \tag{5.7}$$

$$\sum_{t=1}^{T} \pi_t y_t \geq \mu_0, \tag{5.8}$$

$$d_t \geq 0, \qquad t = 1, ..., T, \tag{5.9}$$

$$x_j \geq 0, \qquad j = 1, ..., n, \tag{5.10}$$

Observe that the objective function and the set of constraints (5.5) and (5.9) model the CVaR (see Mansini et. al Mansini et al. (2003) for details), whereas (5.6) gives the expected return in each scenario. Note that, the expected return in each scenario accounts for the net rate of returns, $\sum_{j=1}^{n} r_{jt} x_j$, minus the transaction rates $\sum_{i \in B} p_i x_i$. The sets of constraints (5.7) and (5.10) force x to define a portfolio, and finally constraint (5.8) ensures an expected return of, at least, μ_0.

There are different ways of accounting for the transaction costs in the literature. For instance, including them in the objective function Angelelli et al. (2012); Olivares-

Nadal and DeMiguel (2018); Woodside-Oriakhi et al. (2013), subtracting them from the expected return Krejić et al. (2011); Mansini and Speranza (2005), reducing the capital available for the investment Woodside-Oriakhi et al. (2013), etcetera (see Mansini et al. (2015) and the references therein for further details). Among the different options, as it can be seen in the above formulation, we have incorporated the transaction costs in the investor problem subtracting them from the expected profit.

Note also that by choosing different values for parameters α and μ_0, in the formulation above, different types of investors (i.e. different level of attitude towards risk) can be considered.

5.3 Models description, formulation and resolution algorithms

We consider in this section different forms of structuring the hierarchical relationship.

5.3.1 Bank-Leader Investor-Follower Problem (BLIFP)

We start analyzing the most common hierarchical structure in financial markets in which the bank sets the transaction costs first, and after that, the investor chooses her portfolio. Observe that in this situation, the problem faced from the point of view of the investor reduces to a portfolio selection, under the considered criterion, which in this case is to hedge against risk maximizing the average α-quantile of her smallest returns (CVaR_α). Therefore, we study this situation from the financial intermediary point of view, which is a novel perspective.

We model the situation as a bilevel leader-follower (price setting) problem in which the bank has to fix the transaction costs, from the polyhedral set $\mathbb{P} \in \mathbb{R}^{|B|}$, maximizing its benefits by assuming that, after its decision is made, the investor will make her decision to optimize her considered criterion.

Using the bilevel optimization framework, the **BLIFP** can be modeled as follows:

$$\max \sum_{j \in B} p_j x_j \qquad \textbf{(BLIFP0)}$$

$$\text{s.t. } (5.1), (5.2), (5.3), (5.4), \qquad \text{(Bank Constraints)}$$

$$x \in arg\max \quad \eta - \frac{1}{\alpha} \sum_{t=1}^{T} \pi_t d_t$$

$$\text{s.t. } (5.5), (5.6), (5.7), (5.8), (5.9), (5.10). \qquad \text{(CVaR Constraints)}$$

Our goal is to solve the above problem to provide answers to the new portfolio optimization model. We propose two different MILP formulations with the aim of making a computational comparison to check which one is more effective.

Formulation BLIFP1

The main difficulty in handling (**BLIFP0**) is that some of its decision variables are constrained to be optimal solutions of a nested optimization problem. This nestedness property complicates obtaining a solution of the problem. In order to overcome that issue we observe that the follower problem in (**BLIFP0**) is linear on x when p is given. Hence, as explained in the introductory chapter, this allows us to compute its exact dual as:

$$\min \beta + \mu_0 \mu \qquad (\textbf{Dual1})$$

$$\text{s.t. } \beta - \sum_{t=1}^{T} (r_{jt} - p_j)\delta_t \geq 0, \quad j \in B, \tag{5.11}$$

$$\beta - \sum_{t=1}^{T} r_{jt}\delta_t \geq 0, \qquad j \in R, \tag{5.12}$$

$$-\sum_{t=1}^{T} \gamma_t = 1, \tag{5.13}$$

$$\gamma_t \geq -\frac{\pi_t}{\alpha}, \qquad t = 1, ..., T, \tag{5.14}$$

$$\gamma_t + \delta_t + \pi_t \mu = 0, \qquad t = 1, ..., T, \tag{5.15}$$

$$\gamma_t \leq 0, \qquad t = 1, ..., T, \tag{5.16}$$

$$\mu \leq 0, \beta \geq 0. \tag{5.17}$$

Then, the problem (**BLIFP0**) can be reformulated, applying the Strong Duality Theorem, see Section 1.1.2, including the constraints of the primal and dual problem together with the equation that matches the objective values of the follower primal and dual problems. Thus, (**BLIFP0**) is equivalent to solving this new mathematical programming model:

$$\max \sum_{j \in B} p_j x_j$$

$$\text{s.t. } (5.1), (5.2), (5.3), (5.4), \qquad \text{(Bank Constraints)}$$

$$\eta - \frac{1}{\alpha} \sum_{t=1}^{T} \pi_t d_t = \beta + \mu_0 \mu, \qquad (5.18)$$

$$(5.5), (5.6), (5.7), (5.8), (5.9), (5.10), \qquad \text{(CVaR Constraints)}$$

$$(5.11), (5.12), (5.13), (5.14), (5.15), (5.16), (5.17). \qquad \text{(Dual Constraints)}$$

For the sake of presentation, we have restricted ourselves to consider the problem with only one follower. However, from a theoretical point of view, the problem with several followers will have a similar structure since their decisions are independent. In the model with F followers, there would be F follower problems, then, the number of follower variables and constraints would be multiplied by F. In this situation, the bank's goal would be rather general as maximizing the overall benefit or any other linear function of its prices.

We can observe that in the above formulation we have some bilinear terms, $p_j x_j$ and $p_j \delta_t$ that appear in the leader objective function and constraints (5.6) and (5.11). In order to solve the problem using off-the-shelf solvers, they can be linearized 'a la' McCormick (see McCormick (1976)) giving rise to another exact MILP formulation for the bilevel problem.

Indeed, since $p_j = \sum_{k=1}^{s_j} c_{jk} a_{jk}$, $\forall j \in B$, we could substitute the terms $p_j x_j = \sum_{k=1}^{s_j} c_{jk} \hat{a}_{jk}$ adding variables $\hat{a}_{jk}$, $\forall j \in B, k = 1, ..., s_j$, and the following set of constraints:

$$\begin{array}{ll} \hat{a}_{jk} \leq x_j, & j \in B, k = 1, ..., s_j, \\ \hat{a}_{jk} \leq a_{jk}, & j \in B, k = 1, ..., s_j, \\ \hat{a}_{jk} \geq x_j - (1 - a_{jk}), & j \in B, k = 1, ..., s_j, \\ \hat{a}_{jk} \geq 0, & j \in B, k = 1, ..., s_j. \end{array} \qquad (5.19)$$

Furthermore, this linearization can be simplified. Observe that it is sufficient to include in (**BLIFP0**) the variables $\hat{a}_{jk}$ and the constraints

$$\begin{array}{ll} \hat{a}_{jk} \leq a_{jk}, & j \in B, k = 1, ..., s_j, \\ \hat{a}_{jk} \geq 0, & j \in B, k = 1, ..., s_j, \end{array} \qquad (5.20)$$

from (5.19) and to substitute the variables $x_j = \sum_{k=1}^{s_j} \hat{a}_{jk}, \forall j \in B$. We obtain in this manner an equivalent more compact formulation with the products $a_{jk} x_j$ linearized for all $j \in B, k = 1, ..., s_j$, with less constraints and decision variables.

Following a similar argument we can linearize the products $p_j\delta_t = \sum_{k=1}^{s_j} c_{jk}a_{jk}\delta_t$. To do that, take M a sufficiently large positive number and define the new variables $\hat{\delta}_{jkt} = a_{jk}\delta_t$, $\forall j \in B, k = 1, ..., s_j, t = 1, ..., T$. This set of variables together with the following family of constraints linearize all the bilinear terms:

$$\begin{array}{ll} \hat{\delta}_{jkt} \leq \delta_t, & j \in B, k = 1, ..., s_j, t = 1, ..., T, \\ \hat{\delta}_{jkt} \leq M a_{jk}, & j \in B, k = 1, ..., s_j, t = 1, ..., T, \\ \hat{\delta}_{jkt} \geq \delta_t - (1 - a_{jk})M, & j \in B, k = 1, ..., s_j, t = 1, ..., T, \\ \hat{\delta}_{jkt} \geq 0, & j \in B, k = 1, ..., s_j, t = 1, ..., T. \end{array} \tag{5.21}$$

Combining the above elements, all together, we obtain a valid MILP formulation for **BLIFP**:

$$\max \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk} \qquad \textbf{(BLIFP1)}$$

$$\text{s.t. } \sum_{k=1}^{s_j} a_{jk} = 1, \qquad j \in B, \tag{5.2}$$

$$a_{jk} \in \{0,1\}, \qquad j \in B, k = 1,...,s_j, \tag{5.3}$$

$$p \in \mathbb{P}, \tag{5.4}$$

$$\eta - \frac{1}{\alpha}\sum_{t=1}^{T} \pi_t d_t = \beta + \mu_0\mu \tag{5.18}$$

$$d_t \geq \eta - y_t, \qquad t = 1,...,T, \tag{5.5}$$

$$y_t = \sum_{j\in B} r_{jt}\left(\sum_{k=1}^{s_j}\hat{a}_{jk}\right) + \sum_{j\notin B} r_{jt}x_j - \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk}, \quad t = 1,...,T, \tag{5.22}$$

$$\sum_{j\in B}\sum_{k=1}^{s_j}\hat{a}_{jk} + \sum_{j\notin B} x_j \leq 1, \tag{5.23}$$

$$\sum_{t=1}^{T}\pi_t y_t \geq \mu_0, \tag{5.8}$$

$$d_t \geq 0, \qquad t = 1,...,T, \tag{5.9}$$

$$x_j \geq 0, \qquad j \notin B, \tag{5.10}$$

$$\begin{aligned} &\hat{a}_{jk} \leq a_{jk}, && j \in B, k = 1,...,s_j, \\ &\hat{a}_{jk} \geq 0, && j \in B, k = 1,...,s_j, \end{aligned} \tag{5.20}$$

$$\beta - \sum_{t=1}^{T}\left(r_{jt}\delta_t - \sum_{k=1}^{s_j} c_{jk}\hat{\delta}_{jkt}\right) \geq 0, \quad j \in B, \tag{5.24}$$

$$\beta - \sum_{t=1}^{T} r_{jt}\delta_t \geq 0, \qquad j \in R, \tag{5.12}$$

$$-\sum_{t=1}^{T}\gamma_t = 1, \tag{5.13}$$

$$\gamma_t \geq -\frac{\pi_t}{\alpha}, \qquad t = 1,...,T, \tag{5.14}$$

$$\gamma_t + \delta_t + \pi_t\mu = 0, \qquad t = 1,...,T, \tag{5.15}$$

$$\gamma_t \leq 0, \qquad t = 1,...,T, \tag{5.16}$$

$$\mu \leq 0, \beta \geq 0, \tag{5.17}$$

$$\begin{aligned} &\hat{\delta}_{jkt} \leq \delta_t && j \in B, k = 1,...,s_j, t = 1,...,T, \\ &\hat{\delta}_{jkt} \leq M a_{jk} && j \in B, k = 1,...,s_j, t = 1,...,T, \\ &\hat{\delta}_{jkt} \geq \delta_t - (1 - a_{jk})M && j \in B, k = 1,...,s_j, t = 1,...,T, \\ &\hat{\delta}_{jkt} \geq 0 && j \in B, k = 1,...,s_j t = 1,...,T, \end{aligned} \tag{5.21}$$

The above long formulation can be easily understood once the different sets of constraints are grouped by meaningful blocks. We observe that (5.2), (5.3) and (5.4) are the constraints that define the feasible domain of the bank. Constraint (5.18) imposes the strong duality condition among the primal and dual formulation of the follower problem. Next, (5.5), (5.22), (5.23), (5.8), (5.9), (5.10) and (5.20) are the constraints that correctly define the linearized version of the CVaR subproblem. Finally, the constraints that come from the linearized version of the dual of the follower problem are (5.24),(5.12), (5.13), (5.14), (5.15), (5.16), (5.17) and(5.21).

Using these blocks of constraints Problem (**BLIFP1**) can be written in the following compact form.

$$
\begin{aligned}
\max \sum_{j \in B} \sum_{k=1}^{s_j} c_{jk} \hat{a}_{jk} & \qquad && (\textbf{BLIFP1}) \\
\text{s.t. } (5.2), (5.3), (5.4), & && \text{(Linear Bank Constraints)} \\
(5.18), & && \text{(Strong Duality Constraint)} \\
(5.5), (5.8), (5.9), (5.10), (5.20), (5.22), (5.23), & && \text{(Linear CVaR Constraints 1)} \\
(5.24), (5.12), (5.13), (5.14), (5.15), (5.16), & && \\
(5.17), (5.21). & && \text{(Linear Dual Constraints)}
\end{aligned}
$$

This valid formulation of (**BLIFP1**) requires to set a valid value for the big-M constraint. In the following, we prove the existence of a valid upper bound for such a value.

Proposition 19. *Let $\mathcal{B}(p)$ be the set of all full rank submatrices of the matrix representing the constraints of problem (**Dual1**) in standard form, where p is a fixed set of prices, and let $\mathcal{B}^S(p)$ be the set of all matrices that result from $\mathcal{B}(p)$ replacing, one each time, their columns by the RHS of that problem. Moreover, let $\Delta(p) := \min\{|det(B)| : B \in \mathcal{B}(p)\}$ and $\Delta^S(p) = \max\{|det(B)| : B \in \mathcal{B}^S(p)\}$.*

*Then $UB_\delta := \max_p \Delta^S(p)/\Delta(p)$ is a valid upper bound for the big-M constant in (**BLIFP1**).*

Proof. It is easy to observe that for each fixed set of prices p, $M \leq \max_{t=1,...,T} \delta_t$. Therefore the proof reduces to bound the terms δ_t.

From constraint (5.15) in formulation (**Dual1**) we know that $\delta_t = -\gamma_t - \pi_t \mu, \quad \forall t = 1, ..., T$, which implies that $\delta_t \geq 0$ for all $t = 1, ..., T$, since $\mu \leq 0$, and $\delta_t \leq 0$, and $\pi_t \geq 0$ for all $t = 1, ..., T$.

We observe that $\beta + \mu_0 \mu$ is bounded for any μ_0 and for any set of prices p (recall that this o.f. gives a CVaR) then, if we denote by $r_{max} = \max_{j=1,...,n, t=1,...,T} r_{jt}$, $r_{min} = \min_{j=1,...,n, t=1,...,T} r_{jt}$ and $c_{max} = \max_{j=1,...,n,\ k=1,...,s_j} c_{jk}$, $r_{min} - c_{max} \leq \beta + \mu_0 \mu \leq$

r_{max}. This implies that the solution of (**Dual1**) is attained at an extreme point and therefore no rays have to be considered. Next, the extreme points of the feasible regions are solutions of systems of full dimensional equations taken from the constraint matrix of (**Dual1**) in standard form. Therefore, applying Cramer's rule we obtain that, at the extreme points, the values of any variable δ_t for all $t = 1, \ldots, T$ satisfy: $\delta_t \leq \Delta^S(p)/\Delta(p)$. Next, letting p vary on the finite set of possible prices we obtain that $\delta_t \leq \max_p \Delta^S(p)/\Delta(p)$. □

This bound is only of theoretical interest and in our computational experiments we have set it experimentally to be more accurate.

Formulation BLIFP2

In this section, we derive an alternative formulation for **BLIFP** based on the representation of the prices as $p_j x_j = \sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk}$ in the follower problem before its dual problem is obtained. This artifact produces an alternative compact model that we will analyze in the following.

Let us consider the CVaR problem in (**BLIFP0**), and let us linearize the products of variables $p_i x_i$, as in the previous formulation. This way we obtain:

$$\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t$$

$$\text{s.t. } d_t \geq \eta - y_t, \qquad t = 1, ..., T, \tag{5.5}$$

$$y_t = \sum_{j\in B} r_{jt}\left(\sum_{k=1}^{s_j}\hat{a}_{jk}\right) + \sum_{j\notin B} r_{jt}x_j - \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk}, \quad t = 1, ..., T, \tag{5.22}$$

$$\sum_{j\in B}\sum_{k=1}^{s_j}\hat{a}_{jk} + \sum_{j\notin B} x_j \leq 1, \tag{5.23}$$

$$\sum_{t=1}^{T}\pi_t y_t \geq \mu_0, \tag{5.8}$$

$$d_t \geq 0, \qquad t = 1, ..., T, \tag{5.9}$$

$$x_j \geq 0, \qquad j = 1, ..., n, \tag{5.10}$$

$$\begin{aligned} &\hat{a}_{jk} \leq a_{jk}, && j \in B, k = 1, ..., s_j, \\ &\hat{a}_{jk} \geq 0, && j \in B, k = 1, ..., s_j. \end{aligned} \tag{5.20}$$

Once again, to ease presentation, we write the above formulation in the following compact format.

$$\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t$$
$$\text{s.t. } (5.5), (5.8), (5.9), (5.10), (5.20), (5.22), (5.23). \quad \text{(Linear CVaR Constraints 1)}$$

Its dual problem is:

$$\min \beta + \mu_0\mu + \sum_{j\in B}\sum_{k=1}^{s_j} a_{jk}\sigma_{jk} \qquad \textbf{(Dual2)}$$
$$\text{s.t. } (5.12), (5.13), (5.14), (5.15), (5.16), (5.17),$$
$$\beta - \sum_{t=1}^{T} r_{jt}\delta_t + \sum_{t=1}^{T} c_{jk}\delta_t + \sigma_{jk} \geq 0, \quad j \in B, k = 1, ..., s_j, \tag{5.25}$$
$$\sigma_{jk} \geq 0, \quad j \in B, k = 1, ..., s_j. \tag{5.26}$$

Therefore, we can replace in (**BLIFP0**) the nested optimization problem on the CVaR including the group of constraints in (Linear CVaR Constraints 1) and (5.12)-(5.17), (5.25), (5.26), that we will referred from now on as (Dual2 Constraints), together with the strong duality condition given by

$$\eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t = \beta + \mu_0\mu + \sum_{j\in B}\sum_{k=1}^{s_j} a_{jk}\sigma_{jk}.$$

The combination of all these elements results in the following alternative valid formulation for (**BLIFP0**).

$$\max \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk}$$
$$\text{s.t. } (5.2), (5.3), (5.4) \quad \text{(Bank Constraints)}$$
$$\eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t = \beta + \mu_0\mu + \sum_{j\in B}\sum_{k=1}^{s_j} a_{jk}\sigma_{jk} \tag{5.27}$$
$$(5.5), (5.8), (5.9), (5.10), (5.20), (5.22), (5.23), \quad \text{(Linear CVaR Constraints 1)}$$
$$(5.12), (5.13), (5.14), (5.15), (5.16), (5.17), (5.25), (5.26). \quad \text{(Dual2 Constraints)}$$

The formulation above still contains bilinear terms, namely $a_{jk}\sigma_{jk}$, in constraint (5.27). Therefore, we linearize them as in (**BLIFP1**) and we obtain another valid

MILP formulation for **BLIFP**.

$$\max \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk} \qquad \textbf{(BLIFP2)}$$

$$\text{s.t. } (5.2), (5.3), (5.4) \qquad \text{(Linear Bank Constraints)}$$

$$\eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t = \beta + \mu_0\mu + \sum_{j\in B}\sum_{k=1}^{s_j}\hat{\sigma}_{jk}, \qquad (5.28)$$

$$(5.5), (5.8), (5.9), (5.10), (5.20), (5.22), (5.23), \qquad \text{(Linear CVaR Constraints 1)}$$

$$\begin{array}{l}(5.12),\ (5.13),\ (5.14),\ (5.15),\ (5.16),\ (5.17),\\ (5.25),\ (5.26).\end{array} \qquad \text{(Dual2 Constraints)}$$

$$\begin{array}{ll}\hat{\sigma}_{jk} \le \sigma_{jk}, & j\in B, k=1,...,s_j,\\ \hat{\sigma}_{jk} \le M a_{jk}, & j\in B, k=1,...,s_j,\\ \hat{\sigma}_{jk} \ge \sigma_{jk} - M(1-a_{jk}), & j\in B, k=1,...,s_j\\ \hat{\sigma}_{jk} \ge 0, & j\in B, k=1,...,s_j,\end{array} \qquad (5.29)$$

Again, this valid formulation for (**BLIFP2**) requires to prove the existence of a valid upper bound for the big-M constraint. In the following, we prove that a valid upper bound for such a value does exist.

Proposition 20. *Let UB_δ be the bound obtained in Proposition 19 and $LB_\beta = \min_p \Delta^S(p)/\Delta(p)$. Then $\max\{T(r_{max} - c_{min})UB_\delta - LB_\beta, 0\}$ is a valid upper bound for M in (**BLIFP2**).*

Proof. It is easy to observe that $M = \max_{j\in B, k=1,...,s_j}\{\sigma_{jk}\}$ is a valid upper bound.

Since σ_{jk} is being minimized (it is minimized in (**Dual2**)) and it must satisfy constraints (5.25) and (5.26), there always exists, $\forall j \in B, k = 1, ..., s_j$, an optimal solution where these variables get the values:

$$\sigma_{jk} = \begin{cases} 0, & \text{if } \beta + \sum_{t=1}^{T}(c_{jk} - r_{jt})\delta_t \ge 0\\ -\beta + \sum_{t=1}^{T}(r_{jt} - c_{jk})\delta_t, & \text{otherwise.}\end{cases}$$

Since $\beta \ge 0$ by definition, if it happens that $\beta + \sum_{t=1}^{T}(c_{jk} - r_{jt})\delta_t$ is negative, then $\sum_{t=1}^{T}(c_{jk} - r_{jt}) \le 0$ and therefore $\sum_{t=1}^{T}(r_{jt} - c_{jk}) \ge 0$.

Consequently the maximum value of this variable would be $\max\{0, T(r_{max} - c_{min})UB_\delta - LB_\beta\}$, where UB_δ and LB_β are found by doing a similar discussion as in in the Proposition 19.

□

5.3.2 Investor-Leader Bank-Follower Problem (ILBFP)

For the sake of completeness, in this section, we consider the reverse situation to the one that has been analyzed in Section 5.3.1, i.e. a hierarchical structure in financial markets where the investor acts first and once its portfolio x is chosen the bank sets prices. Although one could claim that this situation may be atypical in actual financial markets, we want to analyse this case from a theoretical point of view. Moreover, we wish to analyse its implications depending of different banks and investors profiles. See Section 5.4 for a comparative analysis. This situation leads to a bilevel leader-follower model in which the investor (leader) has to optimize her utility (maximize the CVaR ensuring a given expected reward, μ_0) by assuming that once she has chosen the portfolio, the bank (follower) will maximize its benefits deciding on the applicable transaction costs.

We can formulate the problem as:

$$\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t \qquad \textbf{(ILBFP0)}$$

$$\text{s.t. } (5.5), (5.6), (5.7), (5.8), (5.9), (5.10), \qquad \text{(CVaR Constraints)}$$

$$p \in arg\max \sum_{j\in B} p_j x_j \qquad (5.30)$$

$$\text{s.t. } (5.1), (5.2), (5.3), (5.4). \qquad \text{(Bank Constraints)}$$

We show in the following proposition that if no further polyhedral constraints are imposed on prices, i.e. $\mathbb{P} = \mathbb{R}_+^{|B|}$, fixing the prices to their maximum possible values is always an optimal solution of the follower (bank) problem.

Proposition 21. *Let **(PricP)** be the follower bank problem, not including constraint (5.4), in the problem **ILBFP0**. Let x be a given portfolio and let $p_j^+ = \max_{k=1,\dots,s_j} c_{jk} \quad \forall j \in B$. Then p_j^+, $\forall j \in B$, is an optimal solution of **(PricP)**.*

Proof. Let us consider **(PricP)** not including constraint (5.4). The solution $p_j^+ \quad \forall j \in B$ is feasible for the problem.

Let x be a given portfolio vector. Clearly, for any given x the objective value given by $\sum_{j\in B} p_j^+ x_j$ can not be improved with any other feasible solution of the bank problem. Hence, this proves the claim. □

Using the previous result, the (**ILBFP0**) can be simplified in the cases in which constraint (5.4) is not included, since the nested optimization problem is replaced by the explicit form of an optimal solution (see Section 1.1.2). This results in a valid linear programming formulation to solve the problem.

$$\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t \qquad \textbf{(ILBFP-LP)}$$

$$\text{s.t. } (5.5), (5.7), (5.8), (5.9), (5.10), \qquad \text{(CVaR Constraints)}$$

$$y_t = \sum_{j=1}^{n} r_{jt}x_j - \left(\sum_{j\in B} p_j^+ x_j\right), \quad t = 1, ..., T.$$

Nevertheless, the above result can not be extended to the case in which a more general polyhedron $\mathbb{P}$ defines the admissible set of transaction costs, and a compact MILP formulation can neither be obtained. With the purpose of solving **ILBFP**, in this more general case, we propose an *'add hoc'* algorithm. To justify its validity we need the following theorem.

Theorem 5. *Let $\vartheta = \sum_{j\in B} p_j x_j$, and denote by Λ the set containing the feasible rates of the bank problem in $\mathbb{P}$. The problem (**ILBFP0**) is equivalent to:*

$$\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T} p_t d_t \qquad \textbf{(ILBFP-Compact)}$$

$$\begin{aligned}
st.\quad & \sum_{j=1}^{n} x_j \leq 1, \\
& d_t \geq \eta - y_t, && t = 1, ..., T, \\
& y_t = \sum_{j=1}^{n} r_{jt}x_j - (\vartheta), && t = 1, ..., T, \\
& \sum_{t=1}^{T} \pi_t y_t \geq \mu_0 \\
& x_j \geq 0, && j = 1, ..., n, \\
& d_t \geq 0, && t = 1, ..., T, \\
& \vartheta \geq \sum_{j\in B} p_{int,j}x_j, && p_{int} \in \Lambda.
\end{aligned}$$

Proof. We prove first that, maximizing the objective function $\eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t$ in (**ILBFP0**) is equivalent to maximizing $\eta(1 - C_x) + \frac{1}{\alpha}\sum_{t\in\mathbb{T}'}\sum_{j=1}^{n}\pi_t r_{jt}x_j - C_x\vartheta$, where $C_x = \sum_{t\in\mathbb{T}'}\frac{\pi_t}{\alpha} > 0$ and $\mathbb{T}' := \{t = 1, ..., n : \eta - y_t \geq 0\}$. Observe that the constraints in (**ILBFP-Compact**) imply that $d_t = \max\{0, \eta - y_t\}$ and $y_t = \sum_{j\in B} r_{jt}x_j -$

$\sum_{j\in B} p_j x_j$ for all $t = 1, ..., T$. Therefore the objective value in the problem satisfies:

$$\begin{aligned}
\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T} \pi_t d_t &= \max \eta - \frac{1}{\alpha}\sum_{t=1}^{T} \pi_t \max\{0, \eta - y_t\} \\
&= \max \eta - \frac{1}{\alpha}\sum_{t\in\mathbb{T}'} \pi_t(\eta - y_t) \\
&= \max \eta(1 - C_x) + \frac{1}{\alpha}\sum_{t\in\mathbb{T}'} \pi_t \left(\sum_{j\in B} r_{jt}x_j - \sum_{j\in B} p_j x_j\right) \\
&= \max \eta(1 - C_x) + \frac{1}{\alpha}\sum_{t\in\mathbb{T}'} \pi_t \left(\sum_{j\in B} r_{jt}x_j\right) - C_x\vartheta. \qquad (5.31)
\end{aligned}$$

Secondly, we have that, for a given portfolio x, the optimal value $\bar{\vartheta}$ of the follower problem is

$$\begin{aligned}
\bar{\vartheta} = \max & \sum_{j\in B} p_j x_j \\
& \text{s.t. } (5.1), (5.2), (5.3), (5.4), \qquad \text{(Bank Constraints)}
\end{aligned}$$

and it is equivalent to evaluate the objective function in all the feasible points and to choose the largest one:

$$\bar{\vartheta} = \max \sum_{j\in B} p_{int,j} x_j, \quad p_{int} \in \Lambda.$$

Since C_x and ϑ are positive, and ϑ is being minimized in (5.31), the follower problem in (**ILBFP0**), can be replaced by

$$\vartheta \geq \sum_{j\in B} p_{int,j} x_j, \ p_{int} \in \Lambda,$$

and the result follows. □

Observe that, if the set of points in Λ were explicitly known, (**ILBFP-Compact**) would be a MILP compact formulation with very likely an exponential number of constraints for the general case of (**ILBFP0**). However, the points in the set Λ are usually difficult to enumerate a priori.

The idea of our algorithm is to start with an incomplete formulation of (**ILBFP-Compact**) and reinforce it with a new inequality, coming from a new point in Λ, after each new iteration of the algorithm.

Algorithm 7

1: **procedure** INITIALIZATION
2: Choose a feasible portfolio x^0. Set $CVaR^0 = +\infty$.
3: **procedure** ITERATION ($\tau = 1, 2, \ldots$)
4: Solve the bank (follower) problem for $x^{\tau-1}$. Let p^τ be an optimal solution.
5: Solve the incomplete formulation:

$$\max \eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t \qquad \textbf{(ILBFP-Incomplete}^\tau\textbf{)}$$

$$\text{st.} \quad \sum_{j=1}^{n} x_j \leq 1,$$

$$d_t \geq \eta - y_t, \qquad t = 1, ..., T,$$

$$y_t = \sum_{j=1}^{n} r_{jt}x_j - \vartheta, \qquad t = 1, ..., T,$$

$$\sum_{t=1}^{T}\pi_t y_t \geq \mu_0$$

$$x_j \geq 0, \qquad j = 1, ..., n,$$

$$d_t \geq 0, \qquad t = 1, ..., T,$$

$$\vartheta \geq \sum_{j\in B} p_j^\nu x_j, \qquad \nu = 1, ..., \tau.$$

6: Let $\chi^\tau = (x^\tau, y^\tau, \eta^\tau, d^\tau)$, and let $(\chi^\tau, \vartheta^\tau)$ be an optimal solution and $CVaR^\tau$ the optimal value.
7: **if** $(\chi^\tau, \vartheta^\tau)$ is feasible in (**ILBFP-Incomplete**$^\tau$) **then**
8: $(\chi^{\tau-1}, p^\tau)$ are optimal solutions of (**ILBFP0**), and $CVaR^\tau$ the optimal value. END.
9: **else if** $(\chi^\tau, \vartheta^\tau)$ is not feasible in (**ILBFP-Incomplete**$^\tau$) **then**
10: go to iteration $\tau := \tau + 1$.

We prove in the following result the optimality of the solution obtained in Algorithm 7 and also the finiteness of it.

Theorem 6. *Algorithm 7 finishes in a finite number of iterations with an optimal solution of (**ILBFP0**).*

Proof. We start guaranteeing the finiteness of the algorithm. On the one hand, the number of feasible solutions of the bank problem is finite, then the number of different cuts $\vartheta \geq \sum_{j\in B} p_j^\tau x_j$ that can be added to the incomplete formulation is also finite. On the other hand, if a repeated cut is added then, $x^{\tau-1}$ is feasible in **ILBFP-Incomplete**$^\tau$, since **ILBFP-Incomplete**$^\tau$ is equal to **ILBFP-Incomplete**$^{\tau-1}$, and then the algorithm stops. Therefore the algorithm finishes in a finite number of iterations.

We continue now proving the optimality of the solution obtained. Let us denote by $CVaR^*$ the optimal value of (**ILBFP0**), that by Theorem 5 is also the optimal value of (**ILBFP-Compact**).

First, assume that $(\chi^{\tau-1}, \vartheta^{\tau-1})$ satisfies the stopping criterion. Then, it is clear that $(\chi^{\tau-1}, \vartheta^{\tau-1})$ is also feasible in **ILBFP-Incomplete**$^\tau$ and $CVaR^\nu \leq CVaR^{\nu-1}$ for all $\nu = 1, ..., \tau$, by construction. Hence, $(\chi^\tau, \vartheta^\tau)$ is also optimal in **ILBFP-Incomplete**$^\tau$ and $CVaR^{\tau-1} = CVaR^\tau$.

Second, we have that $CVaR^* \leq CVaR^\tau$ always holds, since the polyhedron describing the feasible region of (**ILBFP-Compact**) is included in the one defining the feasible region in **ILBFP-Incomplete**$^\tau$.

Finally, we have that if $(\chi^{\tau-1}, p^\tau)$ is feasible in (**ILBFP0**), then $CVaR^* = CVaR^\tau$ and it is an optimal solution of (**ILBFP0**). Therefore, it remains to prove that $(\chi^{\tau-1}, p^\tau)$ is feasible in (**ILBFP0**).

Clearly $\chi^{\tau-1}$ verifies constraints (5.5), (5.7), (5.8), (5.9), (5.10), since they are all included in the incomplete formulation, and also, $x^{\tau-1}$, p^τ verify constraints $p \in arg\max \sum_{j\in B} p_j x_j$, (5.1), (5.2), (5.3) and (5.4), since

$$p^\tau \in arg\max \sum_{j\in B} p_j x_j^{\tau-1}$$
$$\text{s.t. } (5.1), (5.2), (5.3), (5.4). \qquad \text{(Bank Constraints)}$$

To complete the proof we need to check that constraint (5.6) is also satisfied.

Since $p^\tau \in arg\max \sum_{j\in B} p_j x_j^{\tau-1}$, then $\sum_{j\in B} p_j^\tau x_j^{\tau-1} \geq \sum_{j\in B} p_j x_j^{\tau-1}$ for any price p verifying (5.1), (5.2), (5.3) and (5.4). Using the same arguments that in Theorem 5 it follows that variable ϑ is being minimized in **ILBFP-Incomplete**$^\tau$, thus $\vartheta^\tau = \sum_{j\in B} p_j^\tau x_j^{\tau-1}$ and then constraint (5.6) holds.

□

5.3.3 The Maximum Social Welfare Problem (MSWP)

From an economical point of view it is interesting to study the Social Welfare Model in which it is assumed that the financial institution and the investor *cooperate* to improve the social welfare of the society. In fact, apart from the theoretical interest, there may be actual situations in which they may have an incentive to work together to share risk and benefits so as to improve, in this way, their solutions by designing a joint strategy.

We have also analyzed this model for the sake of completeness and to compare the performance of this situation where none of the parties has a hierarchical position over the other one. We think that even if the actual implementation of the cooperative model may be difficult, in a competitive actual market, one may gain some insights into the problem through its analysis.

In this social welfare model we assume that both, financial intermediary and investor, cooperate. If $0 < \xi < 1$ denotes the "proportion" of cooperation of each part, the cooperative version of the problem can be written as a weighted sum of the two objective functions of each party in the feasible region delimited by the constraints of both problems:

$$\begin{aligned} &\max \xi \sum_{j \in B} p_j x_j + (1-\xi)\left(\eta - \frac{1}{\alpha}\sum_{t=1}^{T} \pi_t d_t\right) \\ &\text{s.t. } (5.1), (5.2), (5.3), (5.4), && \text{(Bank Constraints)} \\ &\qquad (5.5), (5.6), (5.7), (5.8), (5.9), (5.10). && \text{(CVaR Constraints)} \end{aligned}$$

The above problem can be modeled as a MILP problem by linearizing the products of variables $a_{jk}x_j, \forall j \in B$ following the same linearization as in Section 5.3.1:

$$\begin{aligned} &\max \xi \sum_{j \in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk} + (1-\xi)\left(\eta - \frac{1}{\alpha}\sum_{t=1}^{T} \pi_t d_t\right) && \textbf{(MSWP0)} \\ &\text{s.t.} \quad (5.2), (5.3), (5.4), && \text{(Linear Bank Constraints)} \\ &\qquad (5.5), (5.8), (5.9), (5.10), (5.20), (5.22), (5.23) && \text{(Linear CVaR Constraints 1)} \end{aligned}$$

Without loss of generality we can consider an unweighted maximum social welfare model where the two objective functions $\sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}a_{jk}$ (bank) and $\eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t$ (investor) are simply added. The following result proves that cooperation is always profitable for both parties in that the joint return exceeds the sum of individual returns of each of them.

Proposition 22. *An optimal solution of the unweighted maximum social welfare model induces an objective value that is greater than or equal to the sum of the optimal returns of the two parties in the same bilevel problem in any of the hierarchical models.*

Proof. Any feasible solution of (**BLIFP0**) and (**ILBFP0**) is feasible in (**MSWP0**) since all the constraints in this last problem appear in the two former formulations. Therefore, the feasible region of (**MSWP0**) includes the feasible regions of both, (**BLIFP0**) and (**ILBFP0**) and the result follows. □

Benders decomposition

We can also obtain a Benders decomposition in order to state a Benders like algorithm to solve (**MSWP0**), and compare the performance of both proposed methods to solve the problem.

Recall that the equal-cooperative model can be written as:

$$\begin{aligned}
\max\ & \sum_{j\in B} p_j x_j + \left(\eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t\right) \\
\text{s.t. } & (5.1), (5.2), (5.3), (5.4), && \text{(Bank Constraints)} \\
& (5.5), (5.6), (5.7), (5.8), (5.9), (5.10). && \text{(CVaR Constraints)}
\end{aligned}$$

In order to apply Benders decomposition we reformulate (**MSWP0**) as follows:

$$\max \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk} + q(y) \qquad \textbf{(MSWP1)}$$

$$\text{s.t.} \quad (5.2), (5.3), (5.4) \qquad \text{(Linear Bank Constraints)}$$

$$\begin{aligned} &\hat{a}_{jk} \le a_{jk}, \quad j \in B, k = 1, ..., s_j, \\ &\hat{a}_{jk} \ge 0, \quad\ \ j \in B, k = 1, ..., s_j, \end{aligned} \qquad (5.20)$$

$$y_t = \sum_{j\in B} r_{jt}\left(\sum_{k=1}^{s_j}\hat{a}_{jk}\right) + \sum_{j\notin B} r_{jt}x_j - \sum_{j\in B}\sum_{k=1}^{s_j} c_{jk}\hat{a}_{jk}, \quad t = 1, ..., T, \qquad (5.22)$$

$$\sum_{j\in B}\sum_{k=1}^{s_j}\hat{a}_{jk} + \sum_{j\notin B} x_j \le 1, \qquad (5.23)$$

$$\sum_{t=1}^{T}\pi_t y_t \ge \mu_0, \qquad (5.8)$$

$$x_j \ge 0, \qquad j \notin B, \qquad (5.10)$$

where

$$\begin{aligned} q(y) = \max\ \ & \eta - \frac{1}{\alpha}\sum_{t=1}^{T}\pi_t d_t \\ \text{s.t.:}\ \ & d_t - \eta \ge -y_t, \qquad t = 1, ..., T, \\ & d_t \ge 0, \qquad\quad t = 1, ..., T. \end{aligned}$$

Note that in $q(y)$ we are essentially computing the CVaR for the given solution $\{y_t : t = 1, \ldots, T\}$.

Computing again its dual problem, the evaluation of $q(y)$ can also be obtained as:

$$\begin{aligned} q(y) = \min & \sum_{t=1}^{T} -\gamma_t y_t \qquad \textbf{(PrimalP)} \\ \text{s.t.:}\ \ & \gamma_t \ge \frac{-\pi_t}{\alpha}, \quad t = 1, ..., T, \\ & -\sum_{t=1}^{T}\gamma_t = 1, \\ & \gamma_t \le 0. \end{aligned}$$

Observe that the above problem, which we define as the Primal Problem, is a

continuous knapsack problem with lower bounds, therefore it is well known that it can be solved by inspection. It suffices to sort non-increasingly the y_t values and assigning, in that order, to each variable γ_t the minimum feasible amount.

Note that in the above formulation the feasible region does not depend on the variables in (**MSWP1**), so if we denote by Λ the set of extreme point solutions of the feasible region of (**PrimalP**), $q(y)$ is equivalent to:

$$\begin{aligned} q(y) = \max \quad & q \\ \text{s.t. } & q \leq \sum_{t=1}^{T} -\gamma_t^{\tau} y_t, \quad \gamma^{\tau} \in \Lambda. \end{aligned} \tag{5.32}$$

Therefore, the problem (**MSWP0**) with discrete prices can be written as:

$$\max \sum_{j \in B} \sum_{k=1}^{s_j} c_{jk} \hat{a}_{jk} + q \qquad \text{(MasterP)}$$

$$\text{s.t.} \quad (5.2), (5.3), (5.4) \qquad \text{(Linear Bank Constraints)}$$

$$\begin{aligned} &\hat{a}_{jk} \leq a_{jk}, \quad j \in B, k = 1, ..., s_j, \\ &\hat{a}_{jk} \geq 0, \quad\ \ j \in B, k = 1, ..., s_j, \end{aligned} \tag{5.20}$$

$$y_t = \sum_{j \in B} r_{jt} \left(\sum_{k=1}^{s_j} \hat{a}_{jk} \right) + \sum_{j \notin B} r_{jt} x_j - \sum_{j \in B} \sum_{k=1}^{s_j} c_{jk} \hat{a}_{jk}, \quad t = 1, ..., T, \tag{5.22}$$

$$\sum_{j \in B} \sum_{k=1}^{s_j} \hat{a}_{jk} + \sum_{j \notin B} x_j \leq 1, \tag{5.23}$$

$$\sum_{t=1}^{T} \pi_t y_t \geq \mu_0, \tag{5.8}$$

$$x_j \geq 0, \qquad j \notin B, \tag{5.10}$$

$$q \leq \sum_{t=1}^{T} \gamma^{\tau} y_t, \quad \gamma^{\tau} \in \Lambda. \tag{5.32}$$

This analysis allows us to state a Benders algorithm as follows:

Algorithm 8 Benders decomposition

procedure INITIALIZATION
  Choose a solution y^0 of the master problem, solve the primal problem (**PrimalP**) for the chosen y^0.
  Let γ^0 be an optimal solution for (**PrimalP**) under y^0 and $q(y^0)$ the corresponding optimal value.
  Take $\boldsymbol{\Upsilon} = \{\gamma^0\}$ and go to iteration $\tau = 1$.
procedure ITERATION ($\tau = 1, 2, \ldots$)
  Solve the master problem (**MasterP**) replacing Λ with Υ. Let y^* and q^* be optimal solutions of such problem.
  if $\tau = 1$ and $q(y^0) = q^*$ **then**
    END.
  else if $\tau > 1$ and $q(y^*) = q^*$ **then**
    END.
  else
    Solve the primal problem (**PrimalP**) for $y = y^*$. Let γ^* be an optimal solution of such problem. Take $\gamma^\tau = \gamma^*$, $\boldsymbol{\Upsilon} = \boldsymbol{\Upsilon} \cup \{\gamma^\tau\}$, and go to iteration $\tau := \tau + 1$.

5.4 Computational Experiment

This section is devoted to report some numerical experiments conducted to: 1) compare the effectiveness of the different methods proposed to solve the different model; 2) analyze the form of the solutions within each model, and 3) compare the profiles of the solutions, in terms of net values for the bank and expected return for the investor, across the three developed models.

The computational experiments were carried out on a personal computer with Intel(R) Core(TM) i7-2600 CPU, 3.40GHz with 16.0 GB RAM. The algorithms and formulations were implemented and solved by using Xpress IVE 8.0.

In order to conduct the computational study, we have considered historical data from IBEX 35. IBEX 35 is the main benchmark stock exchange index of the Spanish stock market. It is made up of the 35 most liquid companies that are listed on the Electronic Spanish Stock Market Interconnection System (SIBE) on the four Spanish stock exchanges (Madrid, Barcelona, Bilbao, and Valencia). We took the monthly returns of these 35 companies during the last three years ($T = 36$ scenarios), and these T historical periods have been considered as equally probable scenarios ($\pi_t = 1/T$).

Different types of instances were generated assuming different random sets B of securities in which the bank can charge some rates. We have considered that this set

can have cardinality ranging in $|B| = 35, 25, 10$. Moreover, we have assumed that the number of different price values s_j, for security $j \in B$, is a random value in the interval $[0, K]$ with $K = 5, 15, 50$. The next table gathers the nine different types of instances (A to I) that we considered:

	$K = 5$	$K = 15$	$K = 50$
$\|B\| = 35$	**A**	**B**	**C**
$\|B\| = 25$	**D**	**E**	**F**
$\|B\| = 10$	**G**	**H**	**I**

Table 5.1: Types of instances depending on the values of $|B|$ and K.

In order to generate each type of instance, different profiles of prices were considered. Approximately 15% of the companies in B were given *cheap* prices; approximately 70% of the companies *normal* prices, and the rest of securities in B, *expensive* prices. Based on actual financial market information, *cheap* prices were generated randomly in the interval $[0.001, 0.003]$, this means that the prices charged by the bank are between the 0.01% and the 0.03% of the invested amount; *normal* prices randomly generated in $[0.002, 0.008]$, and *expensive* prices, generated in the interval $[0.006, 0.010]$.

For each type of instance defined in Table 5.1, five different instances were generated and solved and the average values are reported in all the tables and figures.

With the purpose of making a richer comparison, different profiles of investors with respect to their risk attitude were also considered varying the values of parameters μ_0 and α. We assumed two thresholds for the expected return $\mu_0 = -0.1, 0.0$. This way, we are modeling investors willing to lose, at most, 10% or 0% of their invested amount. In addition, we consider five different CVaR risk levels, $\alpha = 0.1, 0.3, 0.5, 0.7, 0.9$. Note that the smaller the α, the higher the risk-aversion.

5.4.1 Comparing solution methods

This section compares the computational performance of the different methods proposed to solve each one of the models.

For the first model, **BLIFP**, we proposed two different formulations: (**BLIFP1**) and (**BLIFP2**). We show in all our tables, the average CPU time expressed in seconds (CPU) and the number of problems (#) solved to optimality (out of 5) for each formulation, with a time limit of 1800 seconds.

Table 5.2 is organized in two blocks of rows. The first block reports results for $\mu_0 = -0.1$ and the second one for $\mu_0 = 0.0$. Each row in the table refers to a type

of instance $(A, \ldots, I)$. The columns are also organized in five blocks. Each block reports the results for a different risk level α.

It can be observed that (**BLIFP2**) is always faster and it solves a higher number of problems than (**BLIFP1**) to optimality. For example, when $\alpha = 0.1$ and $\mu = 0.0$, (**BLIFP2**) is able to solve all the instances of types B, E, F in at most 165 seconds (on average), while (**BLIFP1**) is not able to solve any of these instances. Therefore, we conclude that formulation (**BLIFP2**) is more effective than (**BLIFP1**) for solving the **BLIFP**, problem.

The second model in our analysis is the one presented in Section 5.3.2, namely **ILBFP**. For this situation, we have proposed a compact LP formulation (**ILBFP-LP**) and the Algorithm 7 to solve the problem. Finally, for **MSWP**, we have also proposed another compact formulation (**MSWP0**) and a Benders' like algorithm (Algorithm 8). The primal problems in the Benders Algorithm 8 were solved by using the inspection method described in the previous section. We report the results concerning these two models in Tables 5.3 and 5.4, and with the same layout as it was already used in Table 5.2. It can be observed that in both models, namely **ILBFP** and **MSWP**, the compact formulations are faster than the algorithms. In spite of that, the algorithms are also able to solve the considered instances quickly. For example, in Table 5.3, the maximum average time spent by Algorithm 7 to solve any instance is almost negligible and less than one second. Analogously, and in table 5.4 the maximum average time spent by the Benders' Algorithm 8 for **MSWP** is less than 11 seconds. Note that in the case of **ILBFP** if additional constraints are impossed over the set of prices, the problem has to be solved with the Benders Algorithms 8.

μ_0		$\alpha = 0.1$				$\alpha = 0.3$				$\alpha = 0.5$				$\alpha = 0.7$				$\alpha = 0.9$			
		(**BLIFP1**)		(**BLIFP2**)		(**BLIFP1**)		(**BLIFP2**)		(**BLIFP1**)		(**BLIFP2**)		(**BLIFP1**)		(**BLIFP2**)		(**BLIFP1**)		(**BLIFP2**)	
		CPU	#	CPU	#	CPU	#	CPU	#	CPU	#	CPU	#	CPU	#	CPU	#	CPU	#	CPU	#
−0.1	**A**	1800	0	1	5	1319	2	1	5	1273	2	2	5	1120	2	2	5	763	3	1	5
	B	1800	0	460	5	1800	0	1238	3	1800	0	1444	1	1799	0	767	3	1800	0	373	4
	C	1800	0	1800	0	1800	0	1800	0	1800	0	1800	0	1800	0	1799	0	1800	0	1474	1
	D	21	5	1	5	729	3	1	5	6	5	2	5	3	5	1	5	2	5	2	5
	E	1451	1	4	5	1800	0	34	5	1392	2	13	5	824	3	7	5	190	5	0	5
	F	1800	0	371	4	1800	0	221	5	1800	0	815	3	1491	2	200	5	1447	1	33	5
	G	2	5	0	5	1	5	0	5	1	5	0	5	1	5	0	5	1	5	0	5
	H	383	5	1	5	408	4	1	5	365	4	1	5	4	5	2	5	2	5	1	5
	I	1121	2	14	5	1472	1	12	5	1138	2	338	5	275	5	1	5	511	4	1	5
0.0	**A**	1153	2	1	5	1467	1	2	5	1370	2	2	5	592	4	1	5	410	5	2	5
	B	1800	0	109	5	1800	0	598	4	1800	0	1095	2	1800	0	784	3	1800	0	447	4
	C	1800	0	1800	0	1800	0	1800	0	1800	0	1800	1	1800	0	1574	1	1800	0	1490	1
	D	16	5	1	5	365	4	2	5	363	4	1	5	5	5	2	5	155	5	2	5
	E	1800	0	2	5	1444	1	16	5	1342	2	13	5	825	3	4	5	33	5	0	5
	F	1800	0	165	5	1800	0	78	5	1800	0	1112	2	1800	0	364	4	1800	0	74	5
	G	4	5	0	5	1	5	0	5	1	5	0	5	1	5	0	5	1	5	0	5
	H	78	5	1	5	17	5	1	5	66	5	1	5	8	5	2	5	721	3	1	5
	I	1638	1	28	5	765	3	19	5	824	3	46	5	51	5	1	5	158	5	1	5

Table 5.2: Comparison of the average CPU and number of problems (out of 5) solved to optimality, for (**BLIFP1**) and (**BLIFP2**).

		$\alpha = 0.1$		$\alpha = 0.3$		$\alpha = 0.5$		$\alpha = 0.7$		$\alpha = 0.9$	
μ_0		**LP**	Alg. 1	**LP**	Alg. 1	**LP**	Alg. 1	**LP**	Alg. 1	**LP**	Alg. 1
−0.1	**A**	0.01	0.07	0.01	0.07	0.01	0.06	0.00	0.07	0.01	0.09
	B	0.00	0.05	0.01	0.09	0.01	0.06	0.00	0.08	0.00	0.10
	C	0.00	0.13	0.04	0.15	0.06	0.38	0.00	0.11	0.01	0.12
	D	0.00	0.07	0.01	0.10	0.00	0.07	0.00	0.06	0.00	0.07
	E	0.00	0.05	0.00	0.10	0.01	0.07	0.01	0.04	0.00	0.08
	F	0.01	0.05	0.00	0.06	0.00	0.05	0.01	0.07	0.00	0.08
	G	0.00	0.05	0.00	0.05	0.01	0.06	0.00	0.04	0.00	0.04
	H	0.00	0.05	0.00	0.05	0.00	0.05	0.01	0.05	0.00	0.05
	I	0.00	0.04	0.00	0.06	0.01	0.05	0.00	0.04	0.00	0.04
0.0	**A**	0.01	0.08	0.00	0.07	0.01	0.08	0.01	0.08	0.00	0.07
	B	0.01	0.09	0.00	0.13	0.01	0.07	0.01	0.08	0.01	0.10
	C	0.01	0.12	0.01	0.17	0.01	0.08	0.01	0.07	0.00	0.12
	D	0.01	0.07	0.00	0.08	0.01	0.05	0.01	0.05	0.01	0.05
	E	0.01	0.10	0.01	0.09	0.01	0.06	0.01	0.05	0.01	0.05
	F	0.01	0.05	0.01	0.10	0.01	0.05	0.01	0.06	0.01	0.07
	G	0.01	0.04	0.00	0.06	0.01	0.04	0.01	0.05	0.01	0.05
	H	0.01	0.05	0.01	0.04	0.01	0.05	0.01	0.05	0.00	0.06
	I	0.01	0.05	0.01	0.05	0.01	0.06	0.01	0.05	0.00	0.04

Table 5.3: Comparison of the average CPU for (**ILBFP-LP**) and Algorithm 7.

		$\alpha = 0.1$		$\alpha = 0.3$		$\alpha = 0.5$		$\alpha = 0.7$		$\alpha = 0.9$	
μ_0		(**MSWP0**)	Ben.	(**MSWP0**)	Ben.	(**MSWP0**)	Ben.	(**MSWP0**)	Ben.	(**MSWP0**)	Ben.
−0.1	**A**	0.03	0.55	0.03	0.79	0.04	1.21	0.03	0.75	0.03	0.47
	B	0.07	0.99	0.07	1.99	0.08	2.83	0.07	1.36	0.06	0.77
	C	0.19	2.63	0.21	4.77	0.21	8.07	0.19	3.94	0.20	2.55
	D	0.02	0.52	0.03	0.92	0.02	1.84	0.02	0.61	0.02	0.52
	E	0.05	0.83	0.06	1.67	0.05	2.42	0.05	1.14	0.05	0.92
	F	0.12	1.74	0.14	3.01	0.14	4.76	0.12	2.45	0.13	1.84
	G	0.01	0.55	0.02	0.77	0.01	1.05	0.01	0.48	0.01	0.34
	H	0.03	0.58	0.03	0.82	0.03	1.29	0.03	0.65	0.03	0.43
	I	0.07	1.01	0.07	1.63	0.07	2.47	0.06	1.40	0.07	0.78
0.0	**A**	0.03	0.74	0.03	5.85	0.06	6.52	0.05	1.18	0.04	0.53
	B	0.10	0.61	0.10	2.48	0.11	3.39	0.10	1.37	0.07	0.86
	C	0.26	1.79	0.27	6.14	0.26	10.50	0.23	3.75	0.20	2.64
	D	0.03	0.45	0.03	0.79	0.03	0.99	0.03	0.50	0.03	0.34
	E	0.09	0.65	0.07	1.89	0.06	3.91	0.07	1.07	0.05	0.73
	F	0.17	1.27	0.16	3.53	0.16	5.01	0.15	2.64	0.14	1.79
	G	0.02	0.36	0.02	0.67	0.02	0.86	0.02	0.41	0.02	0.31
	H	0.03	0.80	0.03	0.86	0.04	1.31	0.03	0.71	0.03	0.36
	I	0.08	0.88	0.09	1.62	0.08	2.40	0.07	1.39	0.06	0.80

Table 5.4: Comparison of the average CPU for (**MSWP0**) and Benders Algorithm 8.

5.4.2 Comparing solutions and risk profiles within models

This subsection analyzes the results provided by the three models in terms of bank net profit and risk and expected return attained by the investor.

Figure 5.1 compares the CVaR values obtained for the different risk profiles. The left subfigure refers to $\mu_0 = -0.1$ whereas the right one to $\mu_0 = 0.0$. Each piecewise curve reports the CVaR values for different α-levels and the nine markets profiles $(A, \ldots, I)$. We observe that in **BLIFP**, the CVaR always increases with the value of α, since this implies to assume more risk. It can also be seen in these figures that, when the value of α increases, the CVaR for $\mu_0 = -0.1$ (left) becomes closer to the CVaR for $\mu_0 = 0.0$ (right). This can be explained because when $\alpha = 1$, if the constraint that the expected return must be greater or equal to 0 is satisfied, both problems become the same, then, the bigger the α the more similar the results for $\mu_0 = -0.1$ and $\mu_0 = 0.0$. Furthermore, for small values of the level α, the CVaR for $\mu_0 = -0.1$ is higher than for $\mu_0 = 0.0$ because the first constraint on the expected return enlarges the feasible region as compared with the second one.

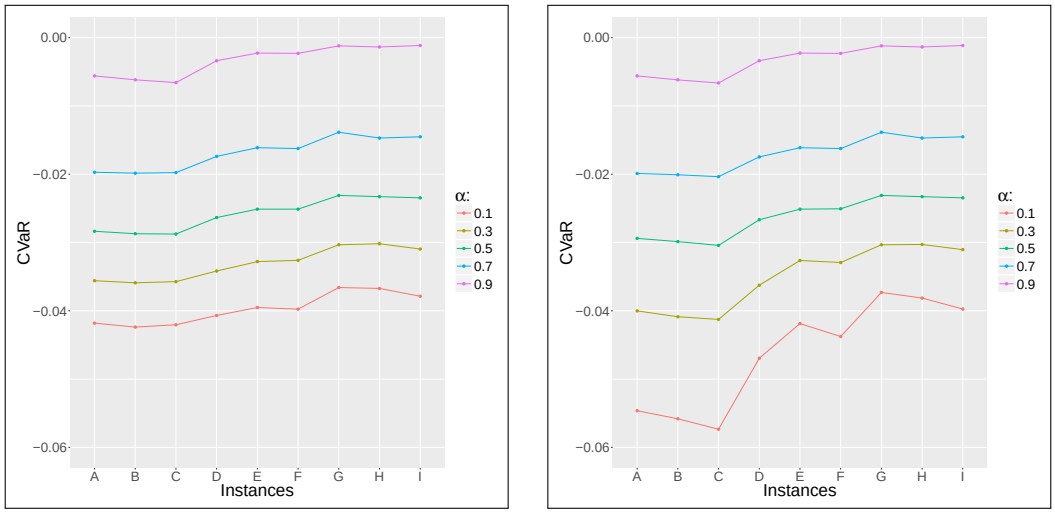

Figure 5.1: Values of the CVaR for model **BLIFP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

Figure 5.2 compares, with a similar organization as in Figure 5.1, the bank net profit for different investor's risk profiles. Analogously, Figure 5.3 represents the expected return for the investor.

We observe in Figure 5.2 that the results of the bank net profit are bigger for profiles with smaller values of α, that is, for more risk-averse investments. In addition, we also show in Figure 5.3 that, in general, bigger expected returns are obtained for

higher values of α. The reason for this is that by increasing α one is considering a wider range of values to compute the CVaR, and then the result is a value closer to the expected return (note that when $\alpha = 1$ the expected return is equal to the CVaR).

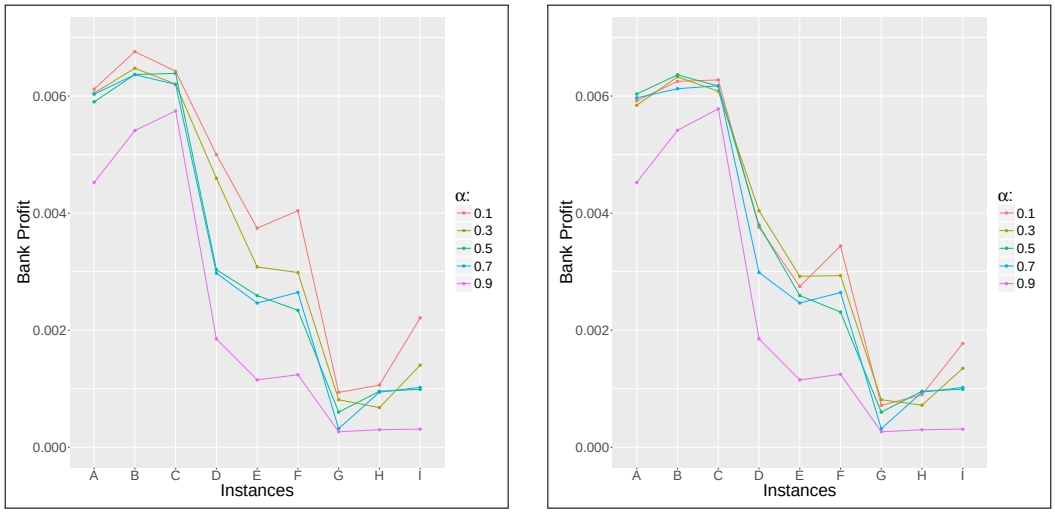

Figure 5.2: Values of the bank profit for model **BLIFP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

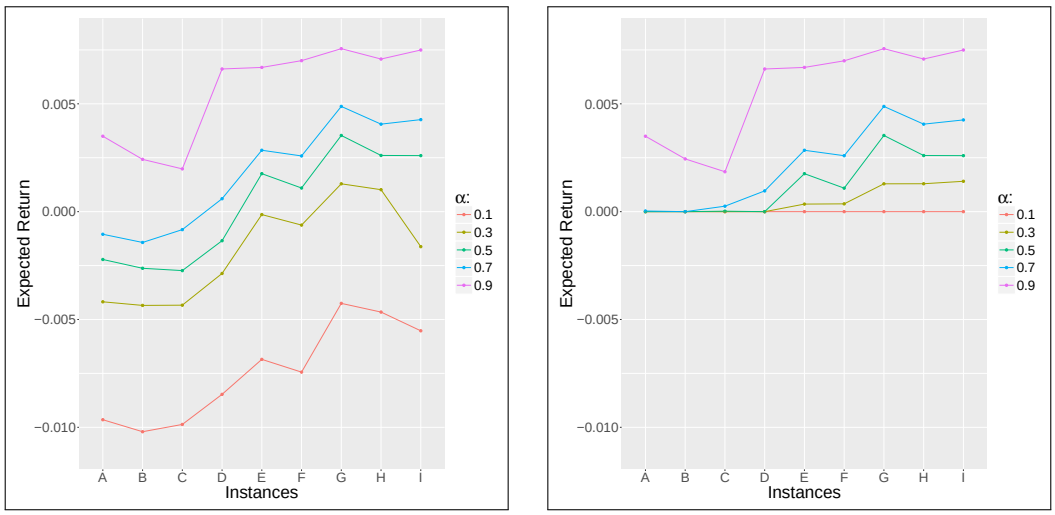

Figure 5.3: Values of the expected return for model **BLIFP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

Finally, to conclude with the analysis of model **BLIFP**, we remark that the smaller the cardinality of the set B the better the CVaR and expected returns for the investor, but the worse the bank net profit. This is clearly expected since we are reducing the number of securities where the bank could charge transaction costs.

We proceed next to analyze the solutions of the second model, namely **ILBFP**. The behavior of these results, Figures 5.4, 5.5 and 5.6, are very similar to those observed in Figures 5.1, 5.2 and 5.3 for the corresponding **BLIFP** model. For instance, we observe in Figure 5.5 the same trend that in the previous model: more risk-averse investments produce bigger profits for the bank, and decreasing the cardinality of the set B results in a reduction of the bank profit.

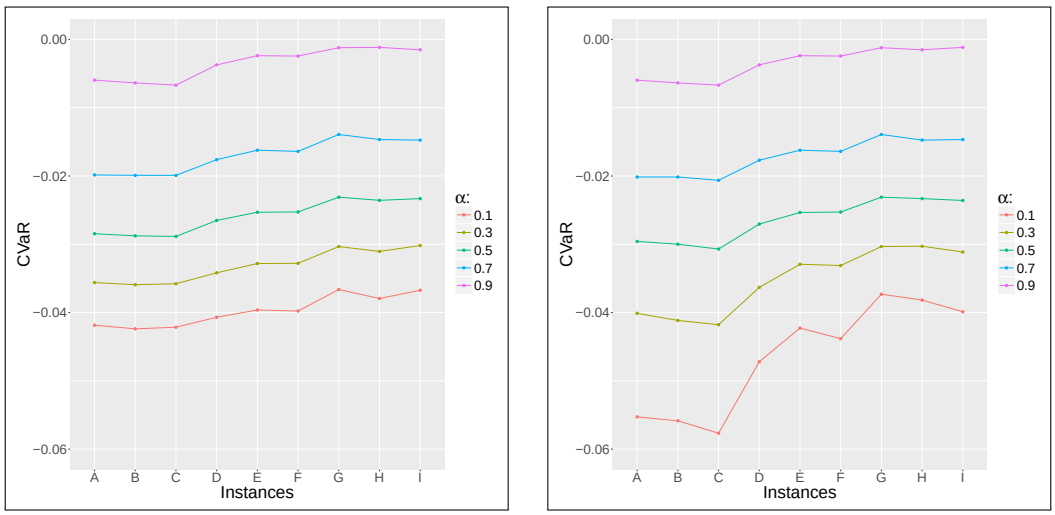

Figure 5.4: Values of the CVaR for model **ILBFP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

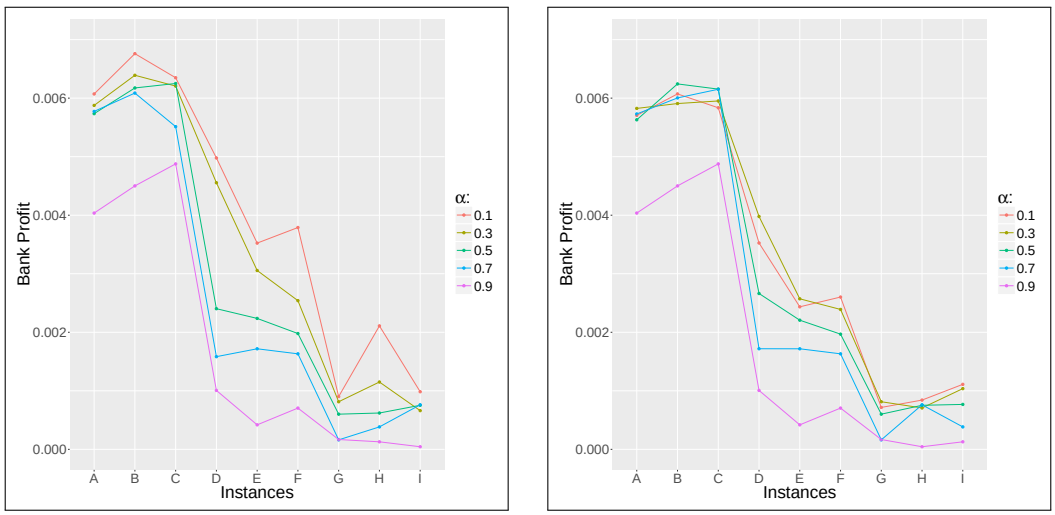

Figure 5.5: Values of the bank profit for model **ILBFP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

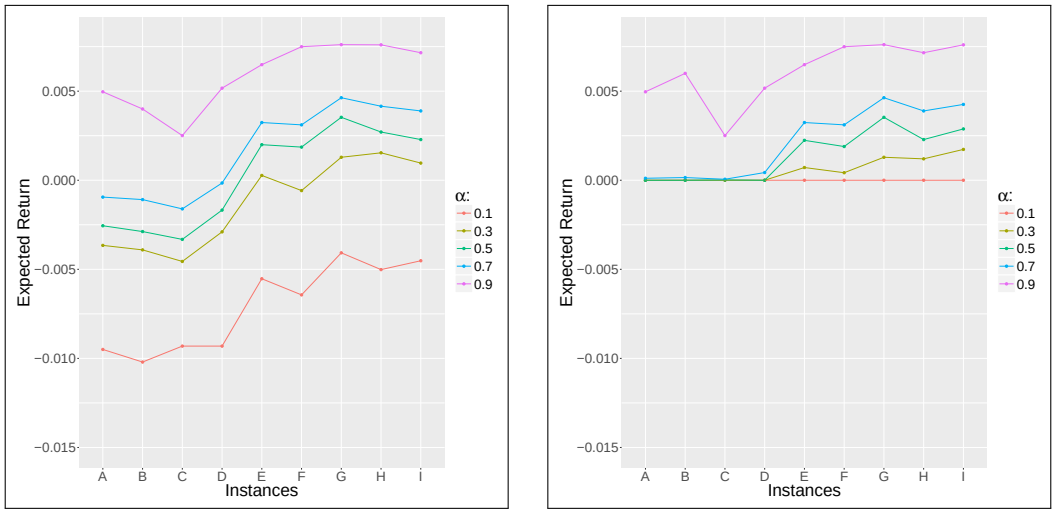

Figure 5.6: Values of the expected return for model**ILBFP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

To finish this section devoted to the analysis of the solutions for our models, we consider the **MSWP** model. In this case, we have only included the comparison of the objective function of this model, namely bank net profit plus CVaR, for the different risk profiles wrt μ_0 and α, and type of market $(A, \ldots, I)$. It can be seen in Figure 5.7 that the sum of the expected profit and the CVaR, i.e. the objective function of the problem, increases with α. One can also observe that for different values of μ_0 the objective values are almost the same. No significant differences are perceived between the different markets for this last model.

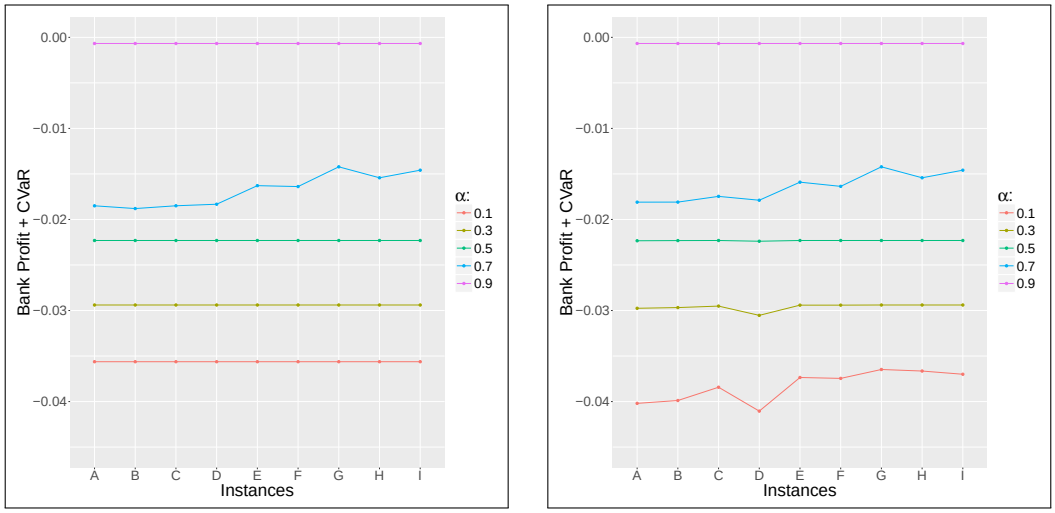

Figure 5.7: Values of the bank profit + CVaR for the model **MSWP**, for different values of α and $\mu_0 = -0.1$ (left), $\mu_0 = 0.0$ (right).

5.4.3 Comparing solutions across models

This last section of the computational experiment is devoted to compare the solutions provided for the three models considered in this chapter, namely **BLIFP**, **ILBFP** and **MSWP**. The goal is to analyze the solution across models with respect to the goals of the two parties: bank net profit, CVaR levels and expected returns.

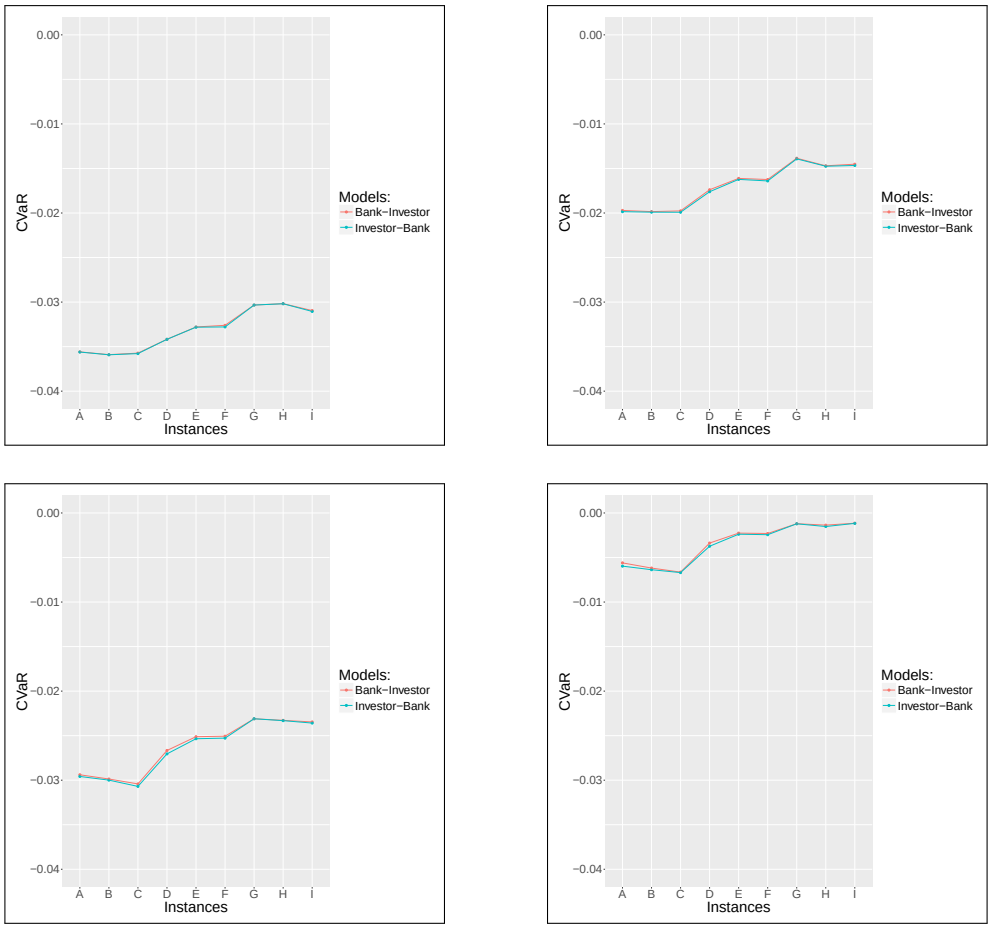

Figure 5.8: Values of the CVaR for the bank-investor model and the investor-bank model, for $\mu_0 = -0.1$, $\alpha = 0.3$ (up-left), for $\mu_0 = -0.1$ $\alpha = 0.7$ (up-right), for $\mu_0 = 0.0$, $\alpha = 0.5$ (down-left) and for $\mu_0 = 0.0$, $\alpha = 0.9$ (down-right).

Figure 5.8 shows a comparison of the CVaR values attained with the **BLIFP** (red line) and **ILBFP** (blue line) for different risk profiles: upper-left $\mu_0 = -0.1$ and $\alpha = 0.3$, upper-right $\mu_0 = -0.1$ and $\alpha = 0.7$, down-left $\mu_0 = 0.0$ and $\alpha = 0.5$ and down-right $\mu_0 = 0.0$ and $\alpha = 0.9$. In all cases, the CVaR values are higher in the **BLIFP** model than in the **ILBFP**. Analogously, Figure 5.9 compares the values of the bank profit for the two models and the same risk profiles. It is also

remarkable that the **BLIFP** model always results in higher profit values for all risk profiles and type of instances. In these comparisons, we do not include the values for the social welfare model because they are not comparable due to the existence of multiple solutions (with the same value for the objective function but very different balance between the distribution of the CVaR and the bank profit). As we mentioned above, we emphasize that in all our experiments the bank-investor model **BLIFP** always gives higher profit for the bank and better CVaR for the investor than the investor-bank model **ILBFP**.

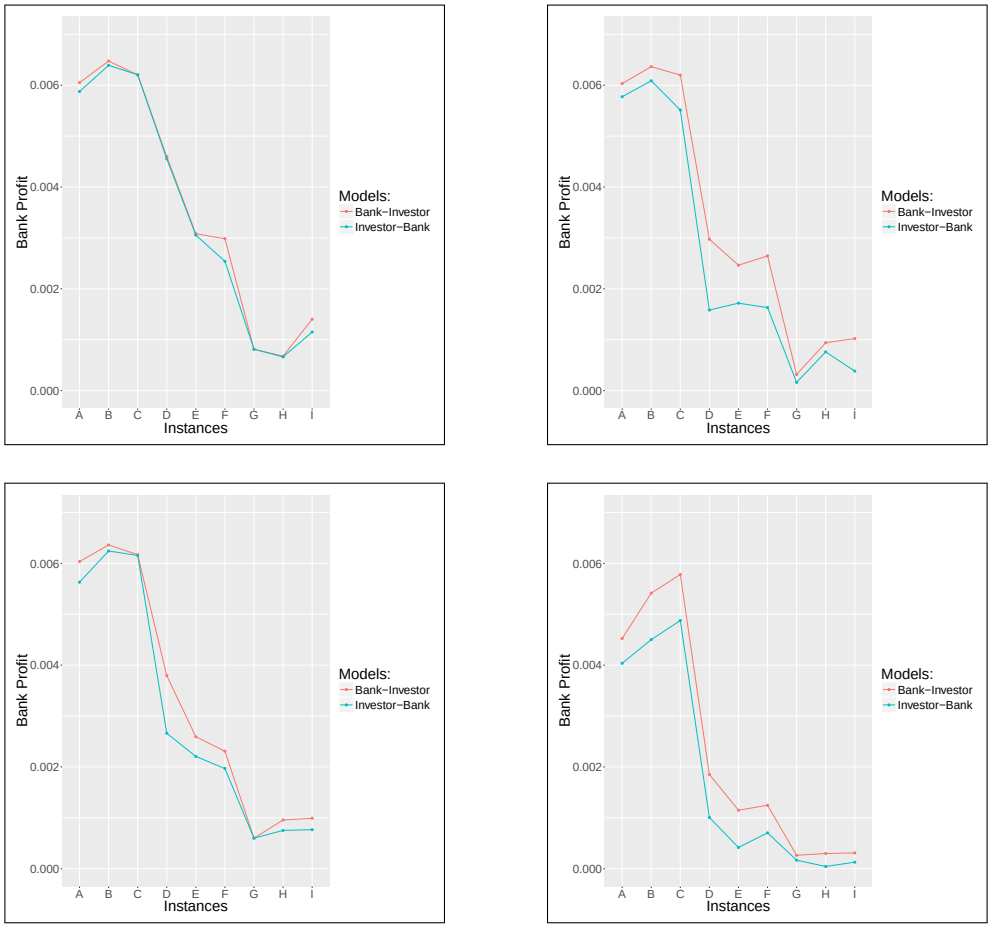

Figure 5.9: Values of the bank profit for the bank-investor model and the investor-bank model, for $\mu_0 = -0.1$, $\alpha = 0.3$ (up-left), for $\mu_0 = -0.1$ $\alpha = 0.7$ (up-right), for $\mu_0 = 0.0$, $\alpha = 0.5$ (down-left) and for $\mu_0 = 0.0$, $\alpha = 0.9$ (down-right).

The last comparisons across models refer to the sum of bank profit + CVaR, in Figure 5.10, and expected return, in Figure 5.11. These two figures show the values attained by the three models for the different instances $(A, \ldots, I)$ and the same four risk profiles that we have already described in the previous cases and in Figures 5.8 and 5.9. As theoretically proved in Proposition 22 the sum of the bank

profit plus the CVaR is always greater for social welfare model **MSWP** than for the other two, namely **BLIFP** and **ILBFP**. To conclude, we compare models with respect to obtained expected returns. Looking at Figure 5.11 for the comparison of expected returns we can not conclude that there exists a model dominating the others with respect to this criterion and therefore the experiments do not prescribe any preference relationship among the models with respect to this element.

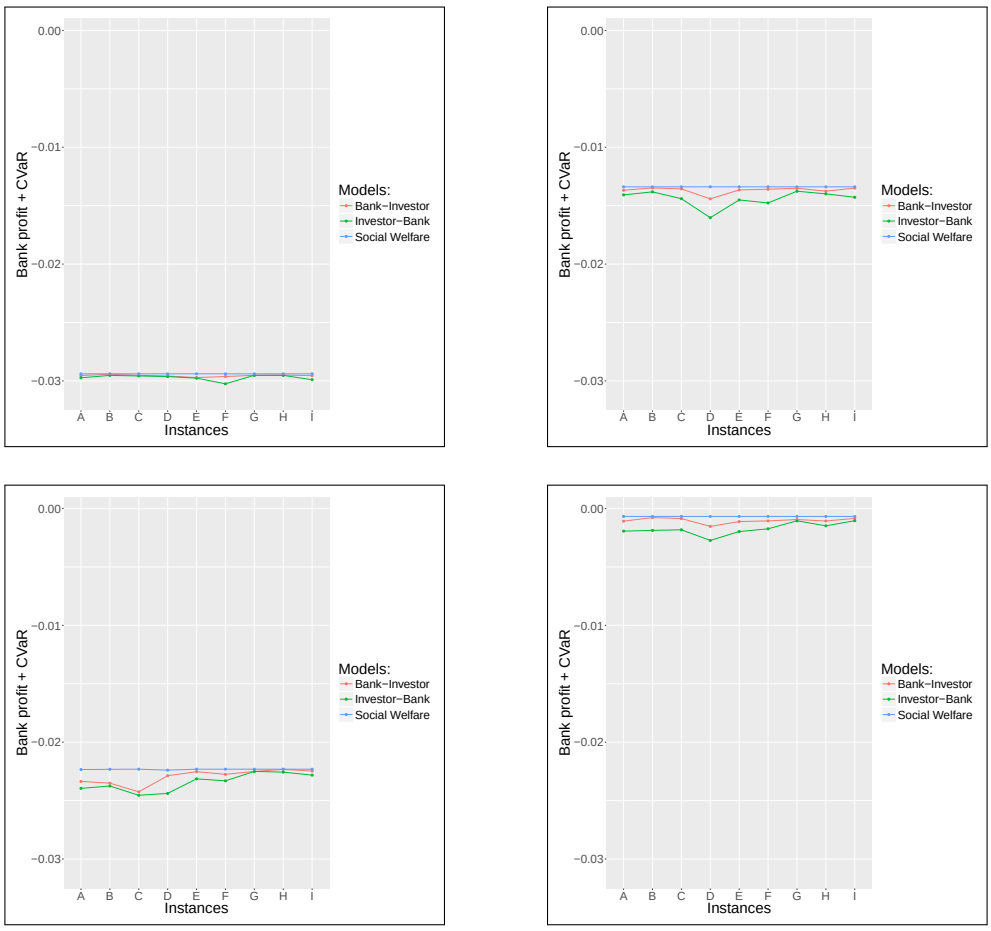

Figure 5.10: Values of the bank profit + CVaR for the three models, for $\mu_0 = -0.1$, $\alpha = 0.3$ (up-left), for $\mu_0 = -0.1$ $\alpha = 0.7$ (up-right), for $\mu_0 = 0.0$, $\alpha = 0.5$ (down-left) and for $\mu_0 = 0.0$, $\alpha = 0.9$ (down-right).

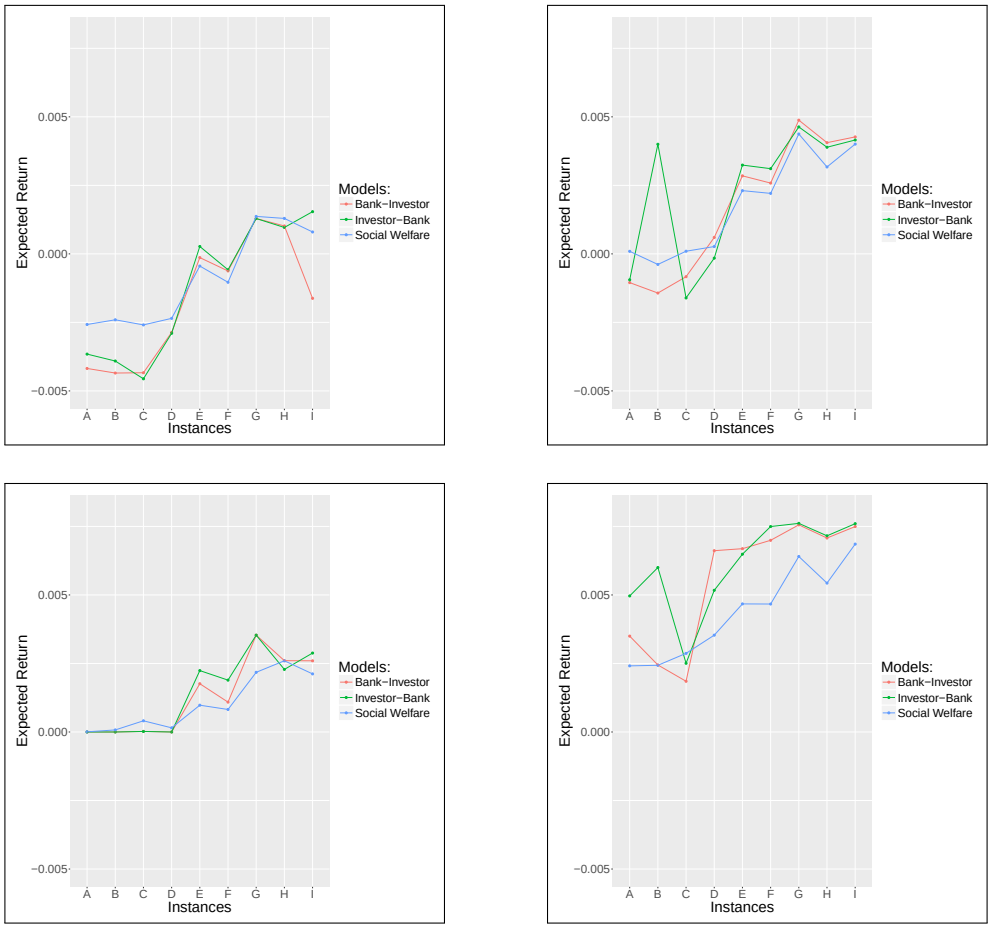

Figure 5.11: Values of the expected return for the three models, for $\mu_0 = -0.1$, $\alpha = 0.3$ (up-left), for $\mu_0 = -0.1$ $\alpha = 0.7$ (up-right), for $\mu_0 = 0.0$, $\alpha = 0.5$ (down-left) and for $\mu_0 = 0.0$, $\alpha = 0.9$ (down-right).

5.5 Conclusions

We have presented a single-period transaction costs portfolio optimization problem with two different decision-makers: the investor and the financial institution. Including the financial intermediaries as decision-makers leads to the incorporation of the transaction costs as decision variables in the portfolio selection problem. The action of both decision-makers was assumed to be hierarchical. We have studied the situations where each party is leader and analyse their implications. This hierarchical structure has been modelled using bilevel optimization. In addition, a social welfare model was also studied.

In all cases, it has been assumed that the financial intermediary had to choose the unit transaction costs, for each security, from a discrete set of prices, maximizing its benefits, and that the investor aimed to minimize the risk (optimizing her CVaR),

ensuring a given expected return. In order to solve the three proposed models, MILP and LP formulations, as well as algorithms, have been proposed. By making variations in the sets of prices, and in the parameters to model the CVaR and the expected return, α and μ_0, different bank and investor profiles can be considered.

In our analysis in Sections 5.3.1 and 5.3.2, all the problems have been presented, for simplicity, with only one follower. Nevertheless, they could be easily extended to more than one. In particular, in Section 5.3.1, the problem has been studied from the point of view of the bank, that is, the bank aimed to maximize its benefit by assuming that once the prices for the securities are set, a single investor will choose her portfolio according to the described goals. We remark that the same procedure could be applied for several followers (investors). In fact, in that model, F different profiles of followers (risk averse, risk taker, etc.) could be considered, and the bank's goal would be maximizing the overall benefit for any linear function of its prices. This approach would allow the bank to improve the decision-making process in the cases where the same prices have to be set for all the investors, but different investor's profiles are considered.

A detailed computational study has been conducted using data from the IBEX 35. We have compared the solution methods, the solutions and the risk profiles within models, and the solutions across models. From our computational experience, we have observed that the bank-leader investor-follower model result in best solutions for both, the bank and the investor, in comparison with the investor-leader bank-follower model, and also that the social welfare model results, as theoretically proved, in higher aggregated benefits.

A future research line could be to extend the model to the case in which the bank considers continuous sets of possible prices. This would lead to hard global optimization models with products of continuous variables. Therefore, other type of techniques must be considered. Also incorporating ordered transaction costs in the pricing could be an interesting research line.

Chapter 6

New bilevel models for the location of *controversial* facilities

In this chapter, motivated by recent real-life applications in Location Theory in which the location decisions generate *controversy*, we propose a novel bilevel location model. In this model, we consider that on the one hand, there is a leader that chooses among a number of fixed potential locations which ones to establish. Next, on the second hand, there is one or several followers that, once the leader location facilities have been set, chooses her location points in a continuous framework. The leader's goal is to maximize some proxy to the weighted distance to the follower's location points, while the follower(s) aim is to locate her location points as close as possible to the leader ones. We develop the bilevel location model for one follower and for any polyhedral distance, and we extend it for several followers and any ℓ_p-norm, $p \in \mathbb{Q}$, $p \geq 1$. We prove the NP-hardness of the problem and propose different mixed integer linear programming formulations. Moreover, we develop alternative Benders decomposition algorithms for the problem. Finally, we report some computational results comparing the formulations and the Benders decomposition on a set of instances.

6.1 Introduction

Location is a research area devoted to the optimal placement of facilities (Albareda-Sambola et al. (2015); Boonmee et al. (2017); Hammad et al. (2018); Kalcsics et al. (2014); Labbé et al. (1995); Nickel and Puerto (2006); Owen and Daskin (1998)), including among many others emergency systems (Bélanger et al. (2019); Boonmee et al. (2017); Schmid and Doerner (2010)), service providers (Albareda-Sambola et al. (2009); Berman et al. (2010)), infrastructures, etcetera, and it is a basic building block of most transportation, communication or logistic problems. Moreover, these models also fit perfectly to positioning problems, as for instance of firms or products, in spaces of attributes or features where the dimension is much higher than in standard location problems on the plane or the tri-dimensional space. An optimal location can be chosen according to different criteria depending on the rationale behind the considered model. The most popular ones are the minimization of the total or maximum transportation cost (Albareda-Sambola et al. (2015); Boonmee et al. (2017)), the maximization of some coverage goal (Albareda-Sambola et al. (2009); Bélanger et al. (2019); Berman et al. (2010); Boonmee et al. (2017)), or the minimization of the undesirable effects induced by the facilities (Brimberg and Juel (1998); Church and Scaparra (2007); Hammad et al. (2018)).

Location Theory includes a number of real-life applications in which the location decisions generate *controversy*. This controversy must be understood as a disagreement among users with different, non-aligned or opposite interests. Examples of this controversial location can be found in the literature, for example, in the areas of

semi-obnoxious facility location or in problems that involve the location and protection of critical infrastructures, the positioning of products in attribute spaces or facilities sensitive to intentional attacks.

In the last decades, the consciousness-raising in environmental issues has grown substantially, specially in those aspects that affect human health or have adverse effects on people. As a consequence of this awareness-raising, the location of (semi-) obnoxious facilities has been extensively studied. Obnoxious facilities are those that generate a disservice to the people nearby while producing an intended product or service (Erkut and Neuman (1989)). However, if only these undesirable effects are taken into account when locating them, these facilities would never be opened or would be located too far from the population centers making use of the produced services, thus generating huge costs. For that reason, in the last years, there has been an increasing focus in analyzing the problem of locating semi-obnoxious facilities (Brimberg and Juel (1998); Hammad et al. (2018); Melachrinoudis and Xanthopulos (2003)). Semi-obnoxious facilities has been defined as useful but unwelcome facilities that produce environmental concerns. That is, facilities that population centers (demand points) want them away, but there are some interests (political, economical ...) in locating them close the demand points, generating in this way, location controversy. Classical examples of this kind of facilities are chemical and power plants, airports, waste dumps, detoxification centers, etc., as listed in Melachrinoudis and Xanthopulos (2003).

Another area that has also attracted increasing attention of researchers in the last years is the location and protection of *vulnerable* facilities (with high risk of disruption) and the protection of critical facilities, including not only those related to disruptions produced by natural disasters or *natural* failures, but also those referred to disruptions produced by man-made attacks (Church and Scaparra (2007); Aksen and Aras (2012); Scaparra and Church (2008)). Critical infrastructure is a term to describe assets that are essential for the functioning of a society and economy. Most commonly associated with the term are facilities for heating, water supply, public health, security services, telecommunication, economic sector, etcetera. Clearly, the location and protection of these types of facilities generates also controversy due to the confrontation between two antagonist parties: *attackers* and *defenders* with visibly opposite goals.

The above-mentioned problems have been usually addressed via biobjective (multiobjective) approaches, difference of objective functions, maximin optimization and, if there exists a hierarchical structure in the decision-making process, by means of bilevel optimization.

Motivated by the increasing interest in real-world applications generating location controversy, we introduce a new model for its study and analysis.

The situation that we want to address models the existence of two parties acting sequentially in a decision-making process. On the one hand, there is a leader who wants to locate some primary facilities and must choose among a number of fixed potential locations where to establish them. On the other hand, there is one (or several) follower(s) that, once the primary facilities have been set, chooses the placement of some secondary facilities, in a continuous environment. The leader's goal is to maximize some proxy of the overall weighted distance to the follower's secondary facility locations. Meanwhile, the follower(s) aim is to locate their secondary facilities as close as possible to the primary ones, minimizing a cost proportional to the distance from the secondary facilities to the primary ones set by the leader.

The reader may observe that this model fits perfectly to the cases mentioned above.

The chapter is structured as follows. The following section introduces the model, sets the notation and proves the NP-hardness of the considered problem. In Section 6.3, we develop the mathematical programming formulations and resolution algorithms for the problem with one follower and any block norm. Two different approaches, based on the representation of the norms, have been considered. Furthermore, due to their importance, they have been applied to the case of the ℓ_1 and ℓ_∞ norm. In the next Section 6.5, we extend the model to several followers and non-polyhedral norms. Section 6.4 is devoted to the computational study of the different methods discussed in the previous sections. Finally, Section 6.6 concludes the chapter.

6.2 Model description

We consider a situation with two different types of location entities: the *primary facilities* (critical infrastructures, goods to protect, demand-points, etc.), and the *secondary facilities* (terrorists nets, thefts, semi-obnoxious facilities, detoxification centers, recycling or power plants, etc.). The primary facilities *wish* to be located as far as possible from the secondary facilities, meanwhile the secondary facilities *aim* to be located as close as possible to the primary ones. The model we present consists in choosing the location of the primary facilities (these are set first), taking into account that, the secondary facilities will be located afterwards knowing their location. For the ease of presentation, we restrict ourselves to the case where a unique secondary facility will be located. The reader is referred to Section 6.5 for the extension to several facilities.

We will model this hierarchical structure using Bilevel Optimization. We assume that there is a leader (setting the primary facilities) that chooses among a set of potential locations B the placement of some new primary facilities. We also consider

that there is a set NB of primary facilities already established, and there exists a budget constraint on the overall investment for the location of the new primary facilities. On the other hand, once the primary facilities are set, the follower chooses the location of the secondary facility in a continuous framework. The proximity between the primary and secondary facilities is measured as a weighted sum of a distance to all primary facilities.

We denote by c_j the cost of opening the primary facility j, for all $j \in B$, by C the maximum budget, by $f_j \in \mathbb{R}^n$ the given location point $j \in B \cup NB$, and by w_j the weight factor that scales the distance from the secondary facility to f_j according to its importance. We define the binary decision variables $y_j = 1$ if f_j, $j \in B$, is open, and $y_j = 0$ otherwise.

For the follower problem we define the decision variable $x \in \mathbb{R}^n$ that specifies the location of the secondary facility.

Therefore, the bilevel problem can be modeled as:

$$\max \sum_{j \in B} w_j d(x, f_j) y_j + \sum_{j \in NB} w_j d(x, f_j) \tag{BLP}$$

$$\text{st.} \sum_{j \in B} c_j y_j \leq C, \tag{6.1}$$

$$y_j \in \{0, 1\} \quad j \in B, \tag{6.2}$$

$$x \in \arg\min_x \sum_{j \in B} w_j d(x, f_j) y_j + \sum_{j \in NB} w_j d(x, f_j),$$

where $d(x, f_j)$ denotes any distance induced by some norm:

$$d(x, f_j) = \|x - f_j\|.$$

Observe that (BLP) is a bilevel max-min problem, in which the objective function of both levels is a proxy of the distance between the primary and the secondary facilities. The resulting bilevel problem contains a knapsack problem at the upper level, and a continuous single-facility location problem at the lower level.

To state the complexity of the problem, we provide the following result.

Theorem 7. *The bilevel location model (BLP) is NP-hard.*

Proof. Let us consider the distance induced by a norm $\|\cdot\|$, and an instance such that $|NB| = 1$ and $w_{j^0} > \sum_{j \in B} w_j$ for $j^0 \in NB$.

For this instance, we know, using the majority theorem, see for example Love et al. (1988), that the optimal solution of the continuous location problem is $x^* = f_j^0$.

Then, if we denote $r_j^* = \|f_j^0 - f_j\|$, Problem (BLP) can be written as:

$$\begin{aligned} &\max \sum_{j\in B} w_j y_j r_j^* \\ &\text{s.t.} \\ &\quad \sum_{j\in B} c_j y_j \leq C, \\ &\quad y_j \in \{0,1\}\ \forall j \in B, \end{aligned}$$

which is a knapsack problem, known to be NP-hard Garey (1979). □

6.3 Model formulations and resolution algorithms

This section is devoted to present useful mathematical programming formulations for (BLP) in order to solve it with of-the-shelf solvers. In addition, we will present alternative add-hoc algorithms, based on decompositions, that prove to be more efficient than the solvers acting on the MILP above-mentioned formulations.

For the sake of presentation, we assume in this section that we measure distances via *block norms*. The family of block norms, also called polyhedral norms, includes all norms $||.||_P$ whose unit ball P is a symmetric with respect to the origin, convex bounded polyhedral set containing the origin in its interior. We will denote then by ext(P) the set of extreme points of P and by P^o : the polar set of P, that is, $P^o := \{x \in \mathbb{R}^n \ : \ \langle x, p\rangle \ \leq 1, \ \forall p \in P\}$. The reader may note that the commonly used ℓ_1 and ℓ_∞ norms belong to this family.

In order to deal with the problem we develop two different procedures: the first one is based on the evaluation of the norm through its primal expression, using its unit ball defined by P, and the second one evaluates the norm through its dual expression, using its dual unit ball P^o.

These two different forms used to handle the problem are justified, as we will see, by the fact that depending on the cases one can be more efficient than the other due to the structure of the set of extreme points of P and P^o. We will illustrate this behavior in the following sections with the ℓ_1 and ℓ_∞ norms.

6.3.1 First approach: Evaluating norms by its primal expression

It is well-known (see for example Nickel and Puerto (2006) and Rockafellar (1970)), that the value of $||x||_P$ is given as:

$$||x||_P = \min \sum_{b \in \text{ext}(P)} \mu_b, \qquad \text{(PrimalNormP)}$$
$$\text{s.t.} \quad x = \sum_{b \in \text{ext}(P)} \mu_b b,$$
$$\mu_b \geq 0, \quad b \in \text{ext}(P).$$

This representation of the norm gives rise to a trilevel problem. Using such representation we develop a MILP Formulation and a Benders like algorithm in order to solve (BLP), using off-the-shelf solvers.

A MILP formulation

Let us assume that $x = (x_1, \ldots, x_n)$, $f_j = (f_{j1}, \ldots, f_{jn})$, for all $j \in B \cup NB$ and $b = (b_1, \ldots, b_n)$ for all $b \in ext(P)$. By representing $||x||_P$ as in (PrimalNormP), (BLP) can be written as the following bilevel problem:

$$\max \sum_{j \in B} w_j y_j r_j + \sum_{j \in NB} w_j r_j \qquad \text{(BLP-P)}$$
$$\text{s.t.} \sum_{j \in B} c_j y_j \leq C, \qquad (6.1)$$
$$y_j \in \{0,1\} \quad j \in B, \qquad (6.2)$$
$$r \in \arg\min_{x,r} \sum_{j \in B} w_j y_j r_j + \sum_{j \in NB} w_j r_j, \qquad (6.3)$$
$$r_j = \sum_{b \in ext(P)} \mu_b^j, \quad j \in B \cup NB, \qquad (6.4)$$
$$x_i = \sum_{b \in ext(P)} \mu_b^j b_i + f_{ji}, \quad j \in B \cup NB, i = 1, ..., n, \qquad (6.5)$$
$$\mu_b^j \geq 0, \quad b \in ext(P), j \in B \cup NB, \qquad (6.6)$$
$$r_j \geq 0, \quad j \in B \cup NB, \qquad (6.7)$$
$$x_i \in \mathbb{R}, \quad i = 1, ..., n. \qquad (6.8)$$

In the above formulation, the secondary facility x is represented in terms of the reference system induced by f_j and the extreme points of P (fundamental directions of the norm $b \in ext(P)$). The summation of the coefficients in this representation gives the norm of a vector (see PrimalNormP). The minimality of the r_j variables is ensured because they are obtained minimizing a linear expression with non-negative coefficients. Therefore, variables r_j, defined in constraint (6.4), represent the distance

between x and f_j and constraints (6.5) set the correct representation of coordinates of the secondary facility in terms of the reference system induced by f_j and the extreme points of P. Constraints (6.1) and (6.2), as in (BLP), are knapsack constraints, corresponding to the choice of the location of the primary facilities, according to a budget constraint. Constraints (6.3)-(6.8) define the lower level problem, the continuous location problem, in which the representation of the norm has been included.

Proposition 23. *Problem (BLP-P) can be reformulated as the following single level problem (BLP-P').*

$$\max \sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j \tag{BLP-P'}$$

$$s.t. \sum_{j\in B} c_j y_j \leq C, \tag{6.1}$$

$$y_j \in \{0,1\} \quad j \in B, \tag{6.2}$$

$$\sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j = \sum_{i=1}^{n} \sum_{j\in B\cup NB} \beta_{ji} f_{ji} \tag{6.9}$$

$$r_j = \sum_{b\in ext(P)} \mu_b^j, \quad j \in B \cup NB, \tag{6.4}$$

$$x_i = \sum_{b\in ext(P)} \mu_b^j b_i + f_{ji}, \quad j \in B \cup NB, i = 1, ..., n, \tag{6.5}$$

$$\mu_b^j \geq 0, \quad b \in ext(P), j \in B \cup NB, \tag{6.6}$$

$$r_j \geq 0, \quad j \in B \cup NB, \tag{6.7}$$

$$x_i \in \mathbb{R}, \quad i = 1, ..., n, \tag{6.8}$$

$$\alpha_j \leq w_j y_j, \quad j \in B, \tag{6.10}$$

$$\alpha_j \leq w_j, \quad j \in NB, \tag{6.11}$$

$$\sum_{j\in B\cup NB} \beta_{ji} = 0, \quad i = 1, ..., n, \tag{6.12}$$

$$-\alpha_j - \sum_{i=1}^{n} b_i \beta_{ji} \leq 0, \quad j \in B \cup NB, b \in ext(P). \tag{6.13}$$

Proof. Given a solution y, representing a feasible set of locations for the primary facilities in (BLP-P), the inner problem in (BLP-P) is a feasible Linear Program (LP) with finite solution, and its dual is:

$$\max \sum_{i=1}^{n} \sum_{j \in B \cup NB} \beta_{ji} f_{ji} \tag{Dual-P}$$

$$\text{s.t. } \alpha_j \leq w_j y_j, \quad j \in B, \tag{6.10}$$

$$\alpha_j \leq w_j, \quad j \in NB, \tag{6.11}$$

$$\sum_{j \in B \cup NB} \beta_{ji} = 0, \quad i = 1, ..., n, \tag{6.12}$$

$$-\alpha_j - \sum_{i=1}^{n} b_i \beta_{ji} \leq 0, \quad j \in B \cup NB, b \in ext(P). \tag{6.13}$$

Then, Problem (BLP-P) is equivalent to the single level formulation (BLP-P') (see Section 1.1.2) since constraint (6.9) is the strong duality condition stating that the primal and dual objectives of the lower level problem must be equal, and the blocks of constraints (6.1)-(6.2), (6.4)-(6.8) and (6.10)-(6.13) represent, respectively, the upper level problem constraints, the lower level primal problem constraints and the lower level dual problem constraints. □

We can observe that the above formulation contains some bilinear terms: $r_j y_j$. In order to transform that formulation into a mixed integer linear problem, the bilinear terms can be linearized 'a la' McCormick (see McCormick (1976)) giving rise to an exact MILP formulation for the bilevel problem. To this end, we substitute the terms $r_j y_j$ by the variables $\hat{r}_j$; $\forall j \in B \cup NB$ and add the following set of constraints:

$$\begin{array}{ll} \hat{r}_j \leq r_j, & j \in B \cup NB, \\ \hat{r}_j \leq M_j y_j, & j \in B \cup NB \\ \hat{r}_j \geq r_j - M_j(1 - y_j), & j \in B \cup NB \\ \hat{r}_j \geq 0, & j \in B \cup NB. \end{array} \tag{6.14}$$

The previous block of constraints requires to set a valid value for the "big-M"-constants. It is easy to observe that M_j can be chosen equal to the maximum distance between f_j and any other point in $B \cup NB$.

Benders like algorithm for solving (BLP).

Now, we propose an alternative method to solve the bilevel location problem under a block norm which is based on a decomposition of the problem.

For a given solution y, the inner problem in (BLP-P) is an LP whose set of constraints does not depend on the variables associated to the master (leader) problem (does not depend on y). Then, if we denote by $\mathbb{P}$ the set of extreme points of the inner problem, solving such problem is equivalent to evaluate the objective function

at the points in $\mathbb{P}$ and to take the minimum objective function value. Then, the continuous location inner problem can be rewritten as the following optimization problem:

$$\begin{aligned} &\max \ q \\ &\text{s.t. } q \leq \sum_{j\in B}\sum_{i=1}^{n} w_j y_j r_j^\tau + \sum_{j\in NB}\sum_{i=1}^{n} w_j r_j^\tau, \qquad \forall\, r^\tau \in \mathbb{P}. \end{aligned}$$

In order to apply Benders decomposition, and using the above formulation, Problem (BLP-P) can be reformulated as:

$$\begin{aligned} &\max\ q \\ &\text{s.t.} \sum_{j\in B} c_j y_j \leq C, \\ &\quad y_j \in \{0,1\}, \quad j \in B, \\ &\quad q \leq \sum_{j\in B}\sum_{i=1}^{n} w_j y_j r_j^\tau + \sum_{j\in NB}\sum_{i=1}^{n} w_j r_j^\tau, \qquad \forall\, r^\tau \in \mathbb{P}. \end{aligned}$$

Again, our approach to solve the above problem is to sequentially identify and add extreme points in $\mathbb{P}$ to the problem until a certificate of optimality is fulfilled (eventually in the worse case after adding all extreme points).

To describe the algorithm, we denote by $\mathcal{P}$ a subset of points in $\mathbb{P}$. With the purpose of obtaining upper bounds for (BLP-P), in the algorithm, we define the following Master Problem:

$$\begin{aligned} &\max \ q \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (\text{MP}) \\ &\text{s.t. } q \leq \sum_{j\in B}\sum_{i=1}^{n} w_j y_j r_j^\tau + \sum_{j\in NB}\sum_{i=1}^{n} w_j r_j^\tau, \qquad r^\tau \in \mathcal{P}, \\ &\quad \sum_{j\in B} c_j y_j \leq C, \\ &\quad y_j \in \{0,1\}\ \forall j \in B. \end{aligned}$$

Lower bounds for the Problem (BLP-P) are obtained in the algorithm by solving the following subproblem:

$$
\begin{aligned}
q(\bar{y}) = \min & \sum_{j \in B} w_j \bar{y}_j r_j + \sum_{j \in NB} w_j r_j, && \text{(PP-P } (\bar{y})\text{)} \\
& r_j = \sum_{b \in ext(P)} \mu_b^j, \quad j \in B \cup NB, \\
& x_i = \sum_{b \in ext(P)} \mu_b^j b_i + f_{ji}, \quad j \in B \cup NB, i = 1, ..., n, \\
& \mu_b^j \geq 0, \quad b \in ext(P), j \in B \cup NB, \\
& r_j \geq 0, \quad j \in B \cup NB, \\
& x_i \in \mathbb{R}, \quad i = 1, ..., n.
\end{aligned}
$$

If $\bar{r}$ is an optimal solution of the above problem for a given solution $\bar{y}$ feasible to the master problem (MP), the inequality $q \leq \sum_{j \in B} \sum_{i=1}^{n} w_j y_j \bar{r}_j + \sum_{j \in NB} \sum_{i=1}^{n} w_j \bar{r}_j$ either generates a new lower bound for (MP) or, if the optimal solution coincides with the previous one, it is a certificate of optimality. Based on this recursion, we propose the following algorithm:

Algorithm 9 Benders decomposition

1: **procedure** INITIALIZATION
2: Choose a solution y^0 satisfying the knapsack constraint, and solve the problem (PP-P $(\bar{y})$) for $\bar{y} = y^0$. Let r^0 be an optimal solution for (PP-P $(\bar{y})$).
3: Take $\mathcal{P} = \{0\}$ and go to iteration $\nu = 1$.
4: **procedure** ITERATION ($\nu = 1, 2, \ldots$)
5: Solve the Master Problem (MP). Let y^* be an optimal solution of such problem and q^* the corresponding optimal value.
6: Solve (PP-P $(\bar{y})$) for $\bar{y} = y^*$.
7: **if** $q^* = q(y^*)$ **then**
8: END.
9: **else**
10: Let r^* be an optimal solution of (PP-P $(\bar{y})$) . Take $r^\nu = r^*$, $\mathcal{P} := \mathcal{P} \cup \{\nu\}$, and go to iteration $\nu := \nu + 1$.

The case of the ℓ_1-norm

In this section, we apply the above reasoning to the particular important case of problem (BLP-P) under the rectangular distance, that is, the distance induced by the ℓ_1-norm. We take advantage of some specific properties of this norm to exploit further its algorithmic implications. As before, n denotes the dimension of the space.

The set of extreme points of the unit ball of the ℓ_1 norm is $ext(P) = \{e_1, ..., e_n, -e_1, ..., -e_n\}$, where e_i the i-th vector of the canonical basis. Further, the ℓ_1-norm of a vector x is given by $\|x\|_1 = \sum_{i=1}^{n} |x_i|$.

By introducing variables r_{ji} representing the non linear terms $|x_i - f_{ji}|$, we adapt (BLP-P) to the ℓ_1-norm case.

$$\max \sum_{j \in B} w_j y_j \sum_{i=1}^{n} r_{ji} + \sum_{j \in NB} w_j \sum_{i=1}^{n} r_{ji} \qquad \text{(BLP}-\ell_1\text{)}$$

s.t.

$$\sum_{j \in B} c_j y_j \leq C,$$

$$y_j \in \{0,1\}, \quad j \in B,$$

$$x \in \arg\min_{x} \sum_{j \in B} w_j y_j \sum_{i=1}^{n} r_{ji} + \sum_{j \in NB} w_j \sum_{i=1}^{n} r_{ji} \qquad (6.15)$$

$$r_{ji} \geq x_i - f_{ji}, \quad j \in B \cup NB, i = 1, ..., n, \qquad (6.16)$$

$$r_{ji} \geq f_{ji} - x_i, \quad j \in B \cup NB, i = 1, ..., n, \qquad (6.17)$$

As in Subsection 6.3.1, we can derive a MILP by using the primal dual optimality conditions and then linearizing the bilinear terms $y_j r_{ji}$ by introducing new variables $\hat{r}_{ji}$. In this formulation, dual variables α_{ji} correspond to contraints (6.16). The dual variables associated to constraints (6.17) have been eliminated.

$$\max \sum_{j\in B} w_j \sum_{i=1}^{n} \hat{r}_{ji} + \sum_{j\in NB} w_j \sum_{i=1}^{n} r_{ji} \qquad \text{(BLP-}\ell_1\text{-1)}$$

$$\text{s.t.} \sum_{j\in B} c_j y_j \leq C, \qquad (6.1)$$

$$y_j \in \{0,1\}, \quad j \in B, \qquad (6.2)$$

$$\sum_{j\in B}\sum_{i=1}^{n} w_j \hat{r}_{ji} + \sum_{j\in NB}\sum_{i=1}^{n} w_j r_{ji} = \sum_{j\in B\cup NB}\sum_{i=1}^{n} -f_{ji}\alpha_{ji} +$$

$$+ \sum_{j\in B}\sum_{i=1}^{n} f_{ji}(w_j y_j - \alpha_{ji}) + \sum_{j\in NB}\sum_{i=1}^{n} f_{ji}(w_j - \alpha_{ji}), \qquad (6.18)$$

$$r_{ji} \geq x_i - f_{ji}, \quad j \in B \cup NB, i = 1, ..., n, \qquad (6.19)$$

$$r_{ji} \geq f_{ji} - x_i, \quad j \in B \cup NB, i = 1, ..., n, \qquad (6.20)$$

$$x_i \in \mathbb{R}, \quad i = 1, ..., n, \qquad (6.8)$$

$$\hat{r}_{ji} \leq M_{ji} y_j, \quad j \in B, i = 1, ..., n, \qquad (6.21)$$

$$\hat{r}_{ji} \leq r_{ji}, \quad j \in B, i = 1, ..., n, \qquad (6.22)$$

$$\hat{r}_{ji} \geq r_{ji} - (1 - y_j)M_{ji}, \quad j \in B, i = 1, ..., n, \qquad (6.23)$$

$$\hat{r}_{ji} \geq 0, \quad j \in B, i = 1, ..., n, \qquad (6.24)$$

$$\alpha_{ji} \leq w_j y_j, \quad j \in B, i = 1, ..., n, \qquad (6.25)$$

$$\alpha_{ji} \leq w_j, \quad j \in B, i = 1, ..., n, \qquad (6.26)$$

$$\alpha_{ji} \geq 0, \quad j \in B \cup NB, i = 1, ..., n, \qquad (6.27)$$

$$\sum_{j\in B} (-2\alpha_{ji} + w_j y_j) + \sum_{j\in NB} (-2\alpha_{ji} + w_j) = 0, \quad i = 1, ..., n. \qquad (6.28)$$

The reader can observe that valid big-M constant in this formulation are $M_{ji} = \max_{k\in B\cup NB} |f_{ki} - f_{ji}|$, for all $i = 1, \ldots, n$ and $j \in B$.

An alternative formulation can be derived for Problem (BLP$-\ell_1$) by using the fact that the inner location problem can be decomposed into n independent linear programs, one for each coordinate. Using the optimality conditions for each such problem and the linearization technique described above, we obtain a formulation (BLP-ℓ_1-2) identical to (BLP-ℓ_1-1) except that constraint (6.18) is replaced by the group of contraints:

$\sum_{j\in B} w_j \hat{r}_{ji} + \sum_{j\in NB} w_j r_{ji} = \sum_{j\in B\cup NB} -f_{ji}\alpha_{ji} + \sum_{j\in B} f_{ji}(w_j y_j - \alpha_{ji}) + \sum_{j\in NB} f_{ji}(w_j - \alpha_{ji}), \quad i = 1, ..., n.$

Algorithm 1 can also be adapted to the case of the ℓ_1-norm. Then, $q(\bar{y})$ is

obtained by solving the lower level problem defined by (6.15) - (6.17) but it can be solved in $O(n|B \cup NB|)$ time since, for each coordinate, it amounts to find the median of a discrete distribution.

We can also use the separability property in the proposed Benders Algorithm, by solving in each iteration n subproblems $q_i(y)$ (one for each coordinate), and considering the following Master Problem:

$$\begin{array}{ll} \max & \sum_{i=1}^{n} q_i \\ \text{s.t.} & \\ & q_i \leq \displaystyle\sum_{j \in B} w_j y_j z_{ij}^{\tau} + \sum_{j \in NB} w_j z_{ij}^{\tau} \quad \forall \tau \in \mathcal{P}, \quad \forall i = 1, ..., n, \\ & \displaystyle\sum_{j \in B} c_j y_j \leq C, \\ & y_j \in \{0,1\} \ \forall j \in B, \end{array} \tag{MP-ℓ_1-i}$$

We will compare the performance of the four approaches in the computational study presented in Section 6.4.

6.3.2 Second approach: Evaluating the norm by its dual expression

Since the polar set of a polyhedron is a polyhedron, P^o induces the so-called dual norm of $||\cdot||_P$ that can also be used to evaluate $||\cdot||_P$. In this case, $||\cdot||_P$ is the optimal solution of the following linear program (see for example Nickel and Puerto (2006) or Rockafellar (1970)):

$$\begin{array}{rl} ||x||_P = \min & r \\ \text{s.t.} & \displaystyle\sum_{i=1}^{n} u_i x_i \leq r, u \in \text{ext}(P^0) \end{array} \tag{NormP^0}$$

Depending on the number and structure of the set of extreme points of P and P^o, it may be more convenient to compute $||\cdot||_P$, by using its primal or dual expression. Further, this dual representation leads to different MILP formulations and the Benders approach can also be adapted.

MILP formulation:

Using the dual representation of the norm, (BLP) can be formulated as:

$$\max \sum_{j \in B} w_j y_j r_j + \sum_{j \in NB} w_j r_j \tag{BLP-P^0}$$

$$\text{s.t.} \sum_{j \in B} c_j y_j \leq C, \tag{6.1}$$

$$y_j \in \{0,1\} \quad j \in B, \tag{6.2}$$

$$r \in \arg\min_{x,r} \sum_{j \in B} w_j y_j r_j + \sum_{j \in NB} w_j r_j, \tag{6.9}$$

$$r_j \geq \sum_{i=1}^{n} u_i (x_i - f_{ji}), \quad u \in ext(P^0), j \in B \cup NB, \tag{6.29}$$

$$r_j \geq 0, \quad j \in B \cup NB, \tag{6.7}$$

$$x_i \in \mathbb{R}, \quad i = 1, ..., n, \tag{6.8}$$

where variables r_j, defined in constraint (6.29), represent the distance between x and f_j. Constraints (6.1) and (6.2) relate the choice of the location of the primary facilities, according to a budget constraint, and constraints (6.9), (6.29), (6.7) and (6.8) define the inner subproblem, in which the representation of the norm via its dual expression has been included.

Proposition 24. *Problem (BLP-P^0) can be formulated as the following single level problem BLP-P^0'.*

$$\max \sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j \qquad \text{(BLP-P}^{0\text{'}}\text{)}$$

$$s.t. \sum_{j\in B} c_j y_j \leq C,$$

$$y_j \in \{0,1\} \quad j \in B,$$

$$\sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j = \sum_{u\in ext(P^0)} \sum_{j\in B\cup NB} (\sum_{i=1}^{n} -f_{ji}u_i)\gamma_{uj},$$

$$r_j \geq \sum_{i=1}^{n} u_i(x_i - f_{ji}), \quad u \in ext(P^0), j \in B \cup NB,$$

$$r_j \geq 0, \quad j \in B \cup NB,$$

$$x_i \in \mathbb{R}, \quad i = 1, ..., n,$$

$$\sum_{u\in ext(P^0)} \gamma_{uj} \leq w_j y_j, \quad j \in B,$$

$$\sum_{u\in ext(P^0)} \gamma_{uj} \leq w_j, \quad j \in NB,$$

$$\sum_{u\in ext(P^0)} \sum_{j\in B\cup NB} (-u_i)\gamma_{uj} = 0, \quad i = 1, ...n,$$

$$\gamma_{uj} \geq 0 \quad u \in ext(P^0), j \in B \cup NB$$

The proof of this proposition follows the same lines as that of Proposition 23 and is thus omitted.

We can observe that in the above formulation there appear the same bilinear terms that we have already obtained in Section 6.3.1. Therefore, the same linearization (6.14) can be applied to obtain the corresponding MILP formulation.

Benders like algorithm for solving (BLP-P^0)

The Benders Algorithm proposed in Section 6.3.1 can also be applied when the norm is induced by the polar polyhedron, with the same Master Problem (MP) and the following primal problem:

$$\begin{aligned}
\min\ & q(y) = \sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j, && \text{(PP-P}^0\text{)}\\
\text{s.t. }\ & r_j \geq \sum_{i=1}^{n} u_{ki}(x_i - f_{ji}), \quad u \in ext(P^0), j \in B \cup NB,\\
& r \geq 0, x \in \mathbb{R}^n,\\
& x_i \in \mathbb{R}, \quad i = 1, ..., n.
\end{aligned}$$

The case of the ℓ_∞-norm

This is Section, we apply the above results to the important case of the infinity norm. The set of extreme points of the infinity norm is $ext(P) = \big\{(a_1, ..., a_n) \in \mathbb{R}^n : a_i \in \{1, -1\}, i = 1, ..n, \big\}$, so that $|ext(P)| = 2^n$. Then formulation (BLP-P') would include $2^n(|B| + |NB|)$ μ_e^j variables, and more than $2^n(|B| + |NB|)$ constraints. However, the number of extreme points of the polar polyhedron is much smaller: $ext(P^0) = \{e_1, ..., e_n, -e_1, ..., -e_n\}$ and $|ext(P^0)| = 2n$. Further, the ℓ_∞-norm of a vector x is given by $\|x\|_\infty = \max_{i=1,...,n} |x_i|$. This allows to adapt (BLP-P) to the ℓ_∞-norm case as follows:

$$\begin{aligned}
\max & \sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j && \text{(BLP}-\ell_\infty\text{)}\\
\text{s.t. } & \sum_{j\in B} c_j y_j \leq C,\\
& y_j \in \{0, 1\}, \quad j \in B,\\
& x \in \arg\min_x \sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j,\\
& r_j \geq x_i - f_{ji}, \quad j \in B \cup NB, i = 1, ..., n,\\
& r_j \geq f_{ji} - x_i, \quad j \in B \cup NB, i = 1, ..., n.
\end{aligned}$$

Further, a MILP formulation can also be derived.

$$\max \sum_{j\in B} w_j\hat{r}_j + \sum_{j\in NB} w_j r_j \qquad \text{(BLP-}\ell_\infty\text{-1)}$$

$$\text{s.t.} \sum_{j\in B} c_j y_j \leq C,$$

$$y_j \in \{0,1\}, \quad j \in B,$$

$$\sum_{j\in B} w_j\hat{r}_j + \sum_{j\in NB} w_j r_j = \sum_{i=1}^{n} \sum_{j\in B\cup NB} f_{ji}(-\gamma_{e_ij} + \gamma_{-e_ij}),$$

$$r_j \geq x_i - f_{ji}, \quad j \in B\cup NB, i = 1,...,n,$$

$$r_j \geq f_{ji} - x_i, \quad j \in B\cup NB, i = 1,...,n,$$

$$r_j \geq 0, \quad j \in B\cup NB,$$

$$\sum_{i=1}^{n} (\gamma_{e_ij} + \gamma_{-e_ij}) \leq w_j y_j, \quad j \in B,$$

$$\sum_{i=1}^{n} (\gamma_{e_ij} + \gamma_{-e_ij}) \leq w_j, \quad j \in NB,$$

$$\sum_{j\in B\cup NB} (-\gamma_{e_ij} + \gamma_{-e_ij}) = 0, \quad i = 1,...,n,$$

$$\gamma_{e_ij} \geq 0, \quad j \in B\cup NB, i = 1,...,n,$$

$$\gamma_{-e_ij} \geq 0, \quad j \in B\cup NB, i = 1,...,n,$$

$$\hat{r}_j \leq My_j, \quad j \in B$$

$$\hat{r}_j \leq r_j, \quad j \in B,$$

$$\hat{r}_j \geq r_j - (1-y_j)M, \quad j \in B,$$

$$\hat{r}_j \geq 0, \quad j \in B.$$

Finally, the proposed Benders Algorithm can also be applied to the problem under the ℓ_∞ norm. The resulting inner subproblem is given by the lower level problem of (BLP$-\ell_\infty$).

6.4 Computational Experiment

In the following we report some numerical results conducted to compare the efficiency of the different methods proposed to solve (BLP), and to check experimentally their scope.

The computational experiments were carried out on a personal computer with Intel B. Core (TM) i7-4720HQ, 2.60 gigahertz with 16384 megabytes RAM. The

MILP formulations and algorithms were implemented and solved by using Xpress Version 8.0.

The distances considered for the numerical experiments were computed using the ℓ_1 and ℓ_∞ norms. Therefore, we implemented the MILP formulations and algorithms proposed in Sections 6.3.1 and 6.3.2, in which we adapted the general methods in Section 6.3 to the models with these two particular distances.

For the computational study we generated different random instances taking into account the following factors: the dimension of the space, n, the cardinality of B and NB, which are the set of potential locations for the new primary facilities and the set of existing primary facilities, respectively, and also the maximum budget C. We considered the following levels for each factor:

- $n = 2, 3, 10, 20$,
- $|B| = 1000, 2000, 5000, 10000$,
- $|NB| = \frac{1}{4}|B|,\ \frac{1}{3}|B|,\ \frac{1}{2}|B|$,
- $C = \frac{1}{C'}|B|$, where $C' = \{3, 4\}$. This C' will be used in the tables for the sake of space.

The costs, c_j, and the weights, w_j, were generated randomly in the interval $[0, 1]$, and each coordinate, f_{ji}, of the location of the primary facilities f_j was generated randomly in the interval $[-1000, 1000]$, for all the instances.

For each combination of levels, 5 different instances were generated and solved. The CPU time limit to solve the problems was set to 1800 seconds.

In Figures 6.1 and 6.2 we show the performance profile graphs of the number of solved instances for the different proposed models for the ℓ_1-norm (Figure 6.1) and ℓ_∞-norm (Figure 6.2). We represent in the abscissa axis the time (in seconds) and in the ordinate axis the number of solved instances. Figure 6.1 reports the results for the ℓ_1-norm and it compares the two MILP formulations (BLP-ℓ_1-1) and (BLP-ℓ_1-2), the basic Benders Algorithm, that we denote by (Bend-ℓ_1), and the Benders algorithm using the separability property, denoted as (Bend-ℓ_1-sep). Figure 6.2 shows the results for the ℓ_∞-norm and it compares the MILP formulation, (BLP-ℓ_∞-1), and Benders algorithm denoted as (Bend-ℓ_∞).

We can observe in Figures 6.1 and 6.2 that the Benders algorithms are more efficient than the MILP formulations, in both cases with the ℓ_1-and-ℓ_∞-norm cases. The Benders algorithms solve all the instances in very short time, whereas none of the MILP formulations could solve to optimality all the instances. We can see in the figures that the formulations (BLP-ℓ_1-1) and (BLP-ℓ_1-2) solve around 300 out

of the 480 instances in 1800 seconds, and formulation (BLP-ℓ_∞-1) solves around 400 instances in the same time.

The average number of cuts added in the Benders algorithm is $5,03$ for (Bend-ℓ_1), $4,75$ for (Bend-ℓ_1-sep) and $4,28$ for (Bend-ℓ_∞). The maximum number of Benders cuts, 14, was added for the (Bend-ℓ_1-sep) for an instance with $n = 10$, $|B| = 5000$, $|NB| = 6667$ and $C = \frac{1}{4}|B|$.

For the ℓ_1-norm case, in Figure 6.1, we can see that the Benders algorithm (Bend-ℓ_1) solves all the instances in approximately 200 seconds, whereas the Benders algorithm using the separability property, (Bend-ℓ_1-sep), needs a bit more time. Nevertheless the performance of both methods is very similar. The same trend can be observed for the MILP formulations, the one without the separability property could solve in the end more instances within the same time limit. However, (BLP-ℓ_1-2) works better for the big instances, $n = 10, 20$, as can bee seen in Figure 6.3, in which we show the performance profile graph of the number of instances solved to optimality by the different proposed models for *big* instances ($n = 10, 20$) and the ℓ_1 norm.

With respect to the ℓ_∞-norm case, Figure 6.2 shows that the Benders algorithm solves all the instances in less than 51 seconds, meanwhile (BLP-ℓ_∞-1) only solves, in the same time, approximately one half of the instances.

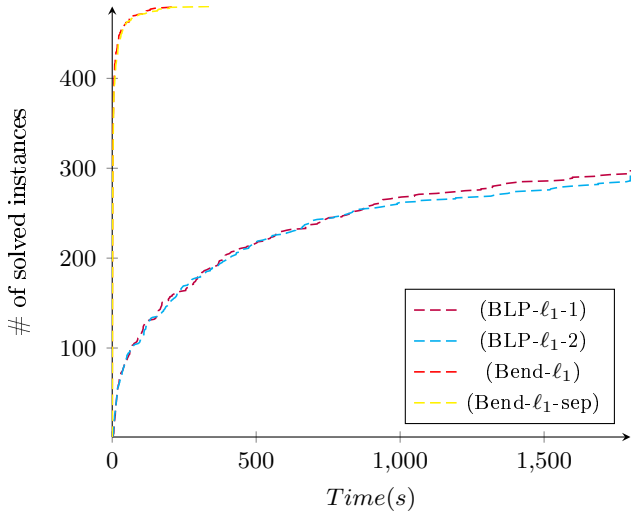

Figure 6.1: Performance profile graph of #solved instances for the different proposed models for the ℓ_1 norm.

More details about the Computational Results of each type of instance can be found in Tables 6.1 and 6.2. We report in this tables the average CPU times (**CPU**), and the numbers of problems, out of 5, solved to optimality (#**OPT**), for each type of instance and each formulation or algorithm. It can be observed, for instance, the speed of the Benders Algorithms in comparison with the MILP formulations, that

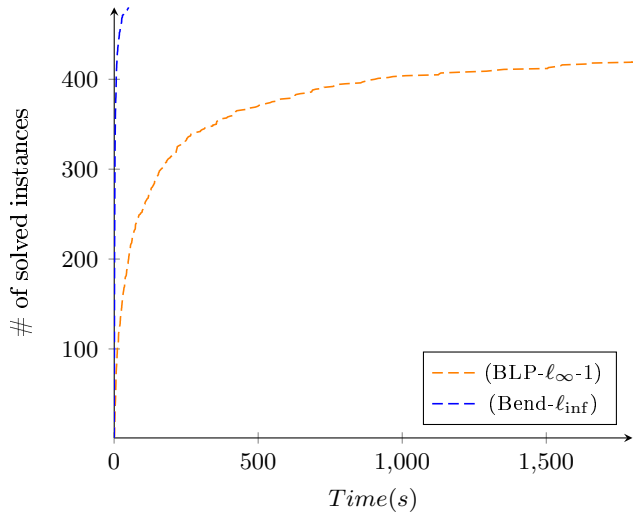

Figure 6.2: Performance profile graph of #solved instances for the different proposed models for the ℓ_∞ norm.

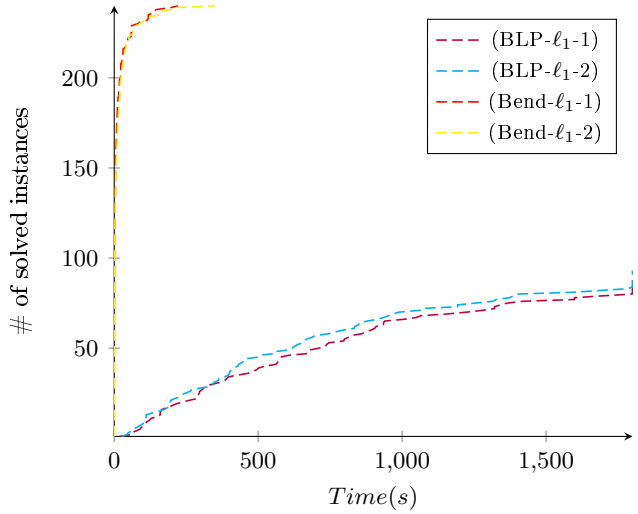

Figure 6.3: Performance profile graph of #solved instances for the different proposed models for *big* instances ($n = 10, 20$) for the ℓ_1 norm.

for example, for the instances of dimension $n = 2$ and $|B| = 10000$, the CPU time for (BLP-ℓ_∞-1) is around 100 seconds meanwhile for the Benders Algorithms around 3 seconds. Or for example, for dimension $n = 3$, $|B| = 10000$ and $|NB| = 5000$, the proposed MILP formulations for (BLP$-\ell_1$) solved few instances within the time limit, whereas the Benders algorithm solved them in less than 6 second. The power of the Benders algorithm becomes even more evident in Table 6.2, since it can be seen that it can solve instances in dimension $n = 20$, and 10000 potential locations in less than 168 seconds.

INSTANCES				(BLP-ℓ_1-1)		(BLP-ℓ_1-2)		(Alg-ℓ_1)		(Alg-ℓ_1-sep)		(BLP-ℓ_∞-1)		(Alg-ℓ_∞)	
n	**|B|**	**|NB|**	**C'**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**
2	1000	250	3	5	10,82	5	10,36	5	0,41	5	0,50	5	2,13	5	0,56
2	1000	250	4	5	4,80	5	5,89	5	0,62	5	0,52	5	2,22	5	1,03
2	1000	333	3	5	13,39	5	10,07	5	0,35	5	0,41	5	2,19	5	0,65
2	1000	333	4	5	7,57	5	4,98	5	0,55	5	0,62	5	2,47	5	1,05
2	1000	500	3	5	11,70	5	12,76	5	0,38	5	0,43	5	8,53	5	0,81
2	1000	500	4	5	7,86	5	6,35	5	0,54	5	0,52	5	3,02	5	0,95
2	2000	500	3	5	36,59	5	37,27	5	0,75	5	1,44	5	6,34	5	0,79
2	2000	500	4	5	13,36	5	11,66	5	0,65	5	0,73	5	9,15	5	1,36
2	2000	667	3	5	217,01	5	152,79	5	0,68	5	0,85	5	7,20	5	0,92
2	2000	667	4	5	20,58	5	14,54	5	0,45	5	0,60	5	6,14	5	1,17
2	2000	1000	3	5	56,48	5	78,63	5	0,45	5	0,61	5	15,29	5	0,81
2	2000	1000	4	5	29,85	5	20,60	5	0,52	5	0,65	5	12,74	5	1,56
2	5000	1250	3	5	204,62	5	508,34	5	1,19	5	1,58	5	48,51	5	1,62
2	5000	1250	4	5	124,68	5	97,10	5	1,49	5	2,36	5	52,24	5	1,70
2	5000	1667	3	5	381,53	5	268,18	5	1,03	5	1,32	5	45,06	5	1,47
2	5000	1667	4	5	135,94	5	198,89	5	2,08	5	1,66	5	37,01	5	1,96
2	5000	2500	3	5	416,01	5	369,43	5	0,80	5	1,12	5	139,15	5	2,34
2	5000	2500	4	5	107,17	5	405,78	5	1,23	5	1,89	5	38,32	5	1,92
2	10000	2500	3	4	708,80	3	955,21	5	1,68	5	2,57	5	137,97	5	3,29
2	10000	2500	4	5	325,03	5	355,51	5	2,44	5	3,75	5	81,66	5	3,72
2	10000	3333	3	5	390,82	5	296,18	5	1,94	5	2,78	5	121,14	5	3,42
2	10000	3333	4	5	329,09	5	439,71	5	3,09	5	3,46	5	159,98	5	3,16
2	10000	5000	3	5	506,03	4	671,22	5	2,16	5	2,09	4	530,49	5	3,38
2	10000	5000	4	5	516,73	5	416,86	5	2,83	5	3,27	5	141,33	5	3,77
3	1000	250	3	5	9,67	5	19,03	5	0,21	5	0,27	5	5,95	5	0,70
3	1000	250	4	5	58,74	5	43,20	5	0,63	5	0,71	5	3,09	5	1,18
3	1000	333	3	5	21,51	5	50,24	5	0,36	5	0,37	5	6,99	5	0,60
3	1000	333	4	5	29,05	5	32,52	5	0,65	5	0,90	5	4,68	5	1,21
3	1000	500	3	5	26,14	5	28,70	5	0,34	5	0,49	5	10,17	5	0,57
3	1000	500	4	5	33,46	5	47,28	5	0,38	5	0,52	5	6,86	5	0,73
3	2000	500	3	5	120,41	5	144,41	5	0,38	5	0,53	5	63,77	5	1,19
3	2000	500	4	5	389,05	5	301,49	5	0,76	5	1,22	5	21,33	5	1,41
3	2000	667	3	5	316,33	5	232,75	5	1,04	5	0,92	5	31,18	5	1,04
3	2000	667	4	5	129,52	5	224,55	5	0,47	5	0,56	5	21,81	5	1,48
3	2000	1000	3	5	113,90	5	261,64	5	0,65	5	0,53	5	81,42	5	1,38
3	2000	1000	4	5	179,31	5	249,42	5	0,62	5	0,71	5	11,05	5	0,88
3	5000	1250	3	4	1014,05	3	1310,24	5	1,95	5	1,85	5	131,33	5	1,94
3	5000	1250	4	3	1352,73	2	1442,08	5	2,43	5	2,24	5	106,38	5	2,12
3	5000	1667	3	4	763,66	4	669,57	5	1,14	5	1,28	5	180,78	5	1,94
3	5000	1667	4	3	1102,02	4	1047,36	5	1,56	5	1,91	5	213,59	5	2,08
3	5000	2500	3	5	636,19	5	689,33	5	0,76	5	1,14	5	391,97	5	2,07
3	5000	2500	4	5	592,53	2	1223,05	5	1,42	5	1,58	5	77,89	5	1,90
3	10000	2500	3	1	1778,05	1	1800,00	5	2,42	5	3,38	5	899,86	5	3,09
3	10000	2500	4	1	1727,79	0	–	5	5,59	5	5,16	5	198,48	5	4,18
3	10000	3333	3	3	1127,45	1	1732,91	5	2,67	5	3,91	5	832,12	5	3,17
3	10000	3333	4	3	1253,69	2	1321,56	5	4,79	5	6,15	4	797,61	5	3,91
3	10000	5000	3	3	1379,12	1	1771,16	5	2,86	5	4,41	4	470,05	5	4,01
3	10000	5000	4	0	–	0	–	5	4,18	5	5,79	5	325,39	5	4,87

Table 6.1: Numerical results for (BLP) under the l_1 and ell_∞ norm.

INSTANCES				(BLP-ℓ_1-1)		(BLP-ℓ_1-2)		(Alg-ℓ_1)		(Alg-ℓ_1-sep)		(BLP-ℓ_∞-1)		(Alg-ℓ_∞)	
n	**|B|**	**|NB|**	**C'**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**	**#OPT**	**CPU**
10	1000	250	3	5	175,69	5	262,45	5	0,52	5	0,84	5	23,53	5	0,73
10	1000	250	4	5	286,71	5	199,67	5	0,58	5	1,02	5	10,04	5	0,60
10	1000	333	3	5	170,27	5	168,75	5	0,29	5	0,57	5	50,58	5	0,68
10	1000	333	4	5	380,23	5	308,91	5	0,79	5	0,98	5	16,31	5	0,66
10	1000	500	3	5	365,20	5	412,48	5	0,56	5	0,64	5	40,66	5	0,80
10	1000	500	4	4	838,00	5	608,46	5	0,74	5	1,17	5	26,14	5	1,16
10	2000	500	3	3	1538,42	3	1310,11	5	0,81	5	1,90	5	104,57	5	1,31
10	2000	500	4	3	1197,88	4	1020,28	5	1,04	5	1,44	5	64,34	5	1,62
10	2000	667	3	2	1406,81	2	1365,28	5	1,23	5	1,20	5	123,13	5	1,22
10	2000	667	4	3	1092,84	4	928,63	5	1,01	5	1,24	5	58,13	5	1,16
10	2000	1000	3	4	1155,24	4	1125,38	5	0,72	5	0,83	5	183,55	5	1,59
10	2000	1000	4	3	980,54	3	1215,99	5	1,11	5	1,84	5	91,77	5	1,66
10	5000	1250	3	1	1673,56	1	1796,25	5	2,66	5	3,00	3	1265,41	5	4,80
10	5000	1250	4	0	–	0	–	5	9,46	5	14,29	5	421,92	5	5,76
10	5000	1667	3	0	–	0	–	5	5,08	5	10,21	4	840,57	5	3,86
10	5000	1667	4	2	1430,78	2	1690,69	5	2,77	5	3,19	4	627,48	5	5,12
10	5000	2500	3	1	1568,60	0	–	5	3,91	5	5,34	4	754,36	5	5,44
10	5000	2500	4	1	1727,35	1	1649,57	5	6,04	5	8,31	3	1125,20	5	5,97
10	10000	2500	3	0	–	0	–	5	9,37	5	17,15	2	1211,56	5	10,24
10	10000	2500	4	0	–	0	–	5	16,19	5	26,46	4	955,69	5	8,48
10	10000	3333	3	0	–	0	–	5	17,88	5	20,49	1	1649,16	5	13,44
10	10000	3333	4	0	–	0	–	5	19,62	5	18,73	3	1152,58	5	10,75
10	10000	5000	3	1	1800,00	0	–	5	11,24	5	23,00	2	1800,00	5	10,91
10	10000	5000	4	0	–	0	–	5	28,00	5	16,26	2	1314,08	5	14,30
20	1000	250	3	4	797,87	5	477,74	5	0,70	5	1,38	5	41,66	5	1,25
20	1000	250	4	5	852,01	5	548,28	5	0,64	5	1,31	5	20,22	5	0,88
20	1000	333	3	4	608,99	5	296,04	5	0,54	5	1,03	5	29,01	5	0,99
20	1000	333	4	4	1026,53	4	773,41	5	1,01	5	1,93	5	39,01	5	1,55
20	1000	500	3	4	881,29	5	412,55	5	0,51	5	1,13	5	55,68	5	1,42
20	1000	500	4	2	1499,86	3	1478,83	5	0,93	5	1,55	5	36,02	5	1,48
20	2000	500	3	1	1629,43	2	1800,00	5	1,93	5	2,21	5	212,07	5	2,08
20	2000	500	4	1	1623,38	2	1634,03	5	3,91	5	2,82	5	217,86	5	2,42
20	2000	667	3	2	1612,20	3	1801,02	5	1,78	5	2,05	5	223,80	5	2,40
20	2000	667	4	3	1421,62	2	1416,62	5	1,34	5	2,44	5	134,67	5	2,64
20	2000	1000	3	0	–	2	1800,00	5	2,45	5	1,86	5	181,95	5	3,02
20	2000	1000	4	1	1800,00	1	1800,00	5	5,09	5	3,08	5	345,70	5	3,44
20	5000	1250	3	0	–	0	–	5	7,20	5	9,37	3	1188,15	5	8,67
20	5000	1250	4	0	–	0	–	5	14,87	5	17,02	5	868,00	5	9,05
20	5000	1667	3	0	–	0	–	5	7,92	5	30,75	2	1513,57	5	8,88
20	5000	1667	4	0	–	0	–	5	7,28	5	18,85	5	1034,49	5	9,46
20	5000	2500	3	0	–	0	–	5	16,23	5	13,53	2	2447,37	5	15,67
20	5000	2500	4	0	–	0	–	5	20,15	5	10,45	2	1585,53	5	13,13
20	10000	2500	3	0	–	0	–	5	52,42	5	65,91	2	1745,86	5	23,08
20	10000	2500	4	0	–	0	–	5	96,02	5	63,30	1	1800,00	5	27,02
20	10000	3333	3	0	–	0	–	5	28,93	5	25,17	2	1750,37	5	22,32
20	10000	3333	4	0	–	0	–	5	125,00	5	93,66	0	–	5	29,79
20	10000	5000	3	0	–	0	–	5	40,39	5	85,61	2	1800,00	5	38,66
20	10000	5000	4	0	–	0	–	5	88,29	5	167,54	1	1800	5	27,76

Table 6.2: Numerical results for (BLP) under the ℓ_1 and l_∞ norm.

6.5 Extensions

This section is devoted to present extensions of the problem (BLP) to several secondary facilities and non-polyhedral norms. We analyze the problem with $K > 1$ secondary facilities which means to locate K new facilities also in the lower level problem. Moreover, we extend the problem (BLP) to deal with norms ℓ_τ for $\tau \in \mathbb{Q}$, $\tau \geq 1$ which requires to apply conic programming and conic duality to obtain results similar to the ones presented in previous sections.

6.5.1 The model with K secondary facilities (independent followers)

We are interested to incorporate to the problem (BLP) the possibility to locate several secondary facilities rather than only one, and the goal of each secondary facility is to minimize the overall distance to the primary facilities. In the following, we analyze problem (BLP) with K secondary points to be located in the lower level problem, that is, we consider that instead of locating one secondary facility, K of these points must be located. For this extension we assume that we are given vectors of weights $w^k \in \mathbb{R}^n_+$, for $k = 1, \ldots, K$, and we define K vectors of decision variables $x^k \in \mathbb{R}^n$, for $k = 1, \ldots, K$; where x^k are the coordinates of the location of the k-th secondary point. With this notation, the new problem can be written as:

$$
\begin{aligned}
&\max \sum_{k \in K} \left(\sum_{j \in B} w_j^k d(x^k, f_j) y_j + \sum_{j \in NB} w_j^k d(x^k, f_j) \right) && \text{(BLP-K)} \\
&\text{s.t.} \sum_{j \in B} c_j y_j \leq C, \\
&\quad y_j \in \{0,1\}, \quad j \in B, \\
&\quad x^k \in \arg\min_{x^k} \sum_{j \in B} w_j^k d(x^k, f_j) y_j + \sum_{j \in NB} w_j^k d(x^k, f_j) \qquad \forall k = 1, ..., K.
\end{aligned}
$$

In the particular case in which $w^1 = w^2 = ... = w^k$, we observe that by symmetry, there is an optimal solution where the secondary facilities co-locate.

Coming back to the general problem (BLP-K), the evaluation of the norm can be done via the primal or dual expression. In both cases, in order to develop a MILP formulation for the model with K secondary facilities, we can apply the same technique that in the previous section. Given a solution y of the upper level problem, the continuous location problem of each follower is linear and thus, the strong duality theorem can be applied as before. This implies that K different one-secondary facility problems are added to the leader problem. In conclusion, the same approach used

with the one-secondary facility location problem is replicated K times and the same results follow.

Furthermore, the Benders Algorithm can also be extended to the case with K followers. The Master Problem for this extension must be slightly modified:

$$\begin{aligned}
\max \quad & \sum_{k\in K} q^k && \text{(MP-K)}\\
\text{s.t.} \quad & q^k \leq \sum_{j\in B}\sum_{i=1}^{n} w_j^k y_j r_j^{k\tau} + \sum_{j\in NB}\sum_{i=1}^{n} w_j^k r_j^{k\tau} && \forall \tau \in \mathcal{P},\ \forall k \in K,\\
& \sum_{j\in B} c_j y_j \leq C,\\
& y_j \in \{0,1\}\ \forall j \in B.
\end{aligned}$$

In addition, in this formulation, there are K primal subproblems with the same structure but with different set of w weights. Therefore, in each iteration of this Benders approach, K primal subproblems must be solved.

6.5.2 The problem under the ℓ_τ-norm

This section extends the analysis of the problem to the case where the inner subproblem measures distances with ℓ_τ-norms via $\tau \in \mathbb{Q},\ \tau \geq 1$. Recall that $||x||_\tau = (\sum_{i=1}^{n} |x_i|^\tau)^{1/\tau}$.

The problem to be considered in this case is

$$\begin{aligned}
\max & \sum_{j\in B} w_j ||x - f_j||_\tau y_j + \sum_{j\in NB} w_j ||x - f_j||_\tau && \text{(BLP-}\ell_\tau\text{)}\\
\text{st.} & \sum_{j\in B} c_j y_j \leq C, && (6.1)\\
& y_j \in \{0,1\} \quad j \in B, && (6.2)\\
& x \in \arg\min_x \sum_{j\in B} w_j ||x - f_j||_\tau y_j + \sum_{j\in NB} w_j ||x - f_j||_\tau,
\end{aligned}$$

Let $\rho \in \mathbb{Q}$ be such that $1/\tau + 1/\rho = 1$.

In order to reformulate Problem **BLP**-ℓ_τ as a single level program, we use standard arguments of conic duality Ye (2004) and a representation for the ℓ_{tau}-norms given in Blanco et al. (2014).

Proposition 25. *The problem **BLP**-ℓ_τ can be reformulated as a single level mixed*

integer conic program.

$$
\begin{aligned}
&\max \sum_{j\in B} w_j||x-f_j||_\tau y_j + \sum_{j\in NB} w_j||x-f_j||_\tau && \textbf{(ConicP)}\\
&s.t.\ \sum_{j\in B} c_j y_j \le C,\\
&y_j \in \{0,1\}, \quad j\in B,\\
&\sum_{j\in B} w_j||x-f_j||_\tau y_j + \sum_{j\in NB} w_j||x-f_j||_\tau = \sum_{j\in B\cup NB}\sum_{i=1}^{n} V_{ji}f_{ji},\\
&x+Z_j = f_j, \quad \forall\, j\in B\cup NB,\\
&||Z_j||_\tau \le r_j, \quad \forall\, j\in B\cup NB,\\
&x\in\mathbb{R}^n,\ Z_j\in\mathbb{R}^n,\ r\in\mathbb{R}^n,\\
&V_{j0} = -w_j y_j, \quad \forall j\in B,\\
&V_{j0} = w_j, \quad \forall j\in NB,\\
&\sum_{j\in B\cup NB} V_{ji} = 0, \quad \forall\, i=1,\ldots,n,\\
&||V_j||_\rho \le V_{j0}, \quad \forall\, j\in B\cup NB,\\
&V_j\in\mathbb{R}^{n+1}, \quad \forall\, j\in B\cup NB.
\end{aligned}
$$

Proof. We observe that the inner location problem can be formulated as the following conic linear program in standard form Luenberger et al. (1984):

$$
\begin{aligned}
&\min \sum_{j\in B} w_j y_j r_j + \sum_{j\in NB} w_j r_j\\
&\text{s.t. } x+Z_j = f_j, \quad \forall\, j\in B\cup NB,\\
&\quad ||Z_j||_\tau \le r_j, \quad \forall\, j\in B\cup NB,\\
&\quad x\in\mathbb{R}^n,\ Z_j\in\mathbb{R}^n,\ r\in\mathbb{R}^n,
\end{aligned}
$$

Therefore, its conic dual can be written, following Ye (2004), as the next formulation:

$$
\begin{aligned}
\max \quad & \sum_{j \in B \cup NB} \sum_{i=1}^{n} V_{ji} f_{ji} \\
\text{s.t.} \quad & V_{j0} = -w_j y_j, \quad \forall j \in B, \\
& V_{j0} = w_j, \quad \forall j \in NB, \\
& \sum_{j \in B \cup NB} V_{ji} = 0, \quad \forall\, i = 1, \ldots, n, \\
& ||V_j||_\rho \leq V_{j0}, \quad \forall\, j \in B \cup NB, \\
& V_j \in \mathbb{R}^{n+1}, \quad \forall\, j \in B \cup NB.
\end{aligned}
$$

Clearly, the inner primal and dual problems satisfy Slater condition so that strong duality applies. This allows us to insert the optimality conditions in **BLP-ℓ_τ** to obtain the final single level program **ConicP**. □

We illustrate the usefulness of the above result with some computational experiments performed on our test instances for the ℓ_2-norm. The model has been implemented in XPRESS 8.0 with the mmnl module that allows solving this type of mixed integer second order cone problems. Table 6.3 summarizes our results. As before, for each combination of n, $|B|$, $|NB|$ and C', we solved 5 different instances and the table reports, in the last two columns, the number of instances out of 5 solved to optimality and the average CPU times

INSTANCES				**ConicP**	
n	$\|B\|$	$\|NB\|$	C'	**#OPT**	**CPU**
2	1000	250	3	5	345,75
2	1000	250	4	4	687,87
2	1000	333	3	3	826,18
2	1000	333	4	4	800,60
2	1000	500	3	4	837,86
2	1000	500	4	5	413,87
2	2000	500	3	2	1235,24
2	2000	500	4	1	1488,53
2	2000	667	3	0	1800,00
2	2000	667	4	1	1632,02
2	2000	1000	3	0	1800,00
2	2000	1000	4	2	1346,31
3	1000	250	3	4	690,83
3	1000	250	4	5	957,83
3	1000	333	3	2	1259,77
3	1000	333	4	2	1429,08
3	1000	500	3	1	1718,03
3	1000	500	4	2	1524,58
3	2000	500	3	0	1800,00
3	2000	500	4	0	1800,00
3	2000	667	3	0	1800,00
3	2000	667	4	0	1800,00
3	2000	1000	3	0	1800,00
3	2000	1000	4	0	1800,00

Table 6.3: Numerical results for **ConicP** under the ℓ_2 norm.

As a result of our tests we have observed that already for $n = 3$ and $|B| = 2000$ none of the instances can be solved to optimality within the time limit which justifies not reporting results for larger size instances.

6.6 Conclusions

This chapter considers models for the location of controversial facilities. Controversial facilities must be understood as those facilities such that their placement induces a disagreement among users with different, non-aligned or opposite interests. Semi-obnoxious facility location and the location and protection of critical infrastructures

or facilities sensitive to intentional attacks are typical examples of this area of research.

We model these situations by a bilevel optimization problem. The first level locates primary facilities trying to be as far away as possible from the secondary ones, which in turns, wish to be as close as possible to the primary ones. We develop mathematical programming formulations for the above mentioned bilevel programs as well as some algorithms that perform very-well in all our experiments that range from small problems on the plane (n=2) with up to $|B| = 10000$, possibilities for the primary facilities until dimension $n = 20$ and $|B| = 10000$.

Chapter 7

Conclusions and future research lines

This thesis has addresses minmax regret and bilevel models, motivated by the increasing trend to incorporate uncertainty sources and multiple decision levels in optimization problems, in the last decades. For all the models we have analyzed, Mathematical Programming techniques have been used to develop MILP formulations and algorithms to solve the problems with off-the-shelf solvers. Some properties of the models have also been studied. Furthermore, the validity of the models has been shown by means of practical applications, and detailed computational experiments have been conducted to check the performance, scope, and speed of the proposed algorithms and formulations. Next, we briefly summarize the major achievements and discuss possible future research lines of each chapter.

In Chapter 2, we have proposed a model to build robust supply networks under uncertainty affecting the activity. A scenario-based representation has been used to model the uncertainty, which can be useful to find a network design in unstable systems or when no historical data about the parameters is available, as for instance, in humanitarian logistics in emergency situations. A Benders decomposition framework has been proposed to solve the models, which has shown to be applicable for different sources of uncertainty. However, for *big* networks, certifying optimality could be intractable. Hence, an interesting research line may be addressed to reinforce the algorithm by initializing it with network designs close to optimal ones, or by developing other types of cuts. Furthermore, another future research line could be related with the extension of the model to more general supply chain management problems, considering hence other sources of uncertainty, as for example, the dependence of the quality of the transported goods on the travel time.

Motivated by real life applications, as scheduling, task sequencing, or grid computing, in Chapter 3, we have studied minmax regret Shortest Path and Traveling Salesman Problems in which the arc costs depend on their relative position on the given path and there exist uncertain cost parameters. In order to model the time-dependent Shortest Path Problem, we have extended the classical development to formulate minmax regret problems with hypercube uncertainty sets, by defining linear functions of the feasible solutions to obtain the maximum regret. The more realistic case, in which dependency relations among all the parameters are allowed, has also been considered for the TDSPP and the TDTSP; for which we have proposed different algorithms based on Benders cuts. The one based on a combination of the other algorithms seemed to work better, being able to solve a bigger number of instances in the computational study. The algorithms have been reinforced with approximation cuts from mid-point scenarios for the hypercube uncertainty set case. Hence, a possible future research line could be to develop approximation results for more general uncertainty sets. Furthermore, as the computational experiment reveals, the increase in the size of the problem makes more difficult the resolution

of it due to the use of time-expanded networks; then, the study of preprocessing techniques could be an interesting and useful element to be analyzed in the future.

In Chapter 4, as a novelty in the minmax regret literature, we have presented a model in which the hypercube of scenarios, S_t, for the uncertain parameters varies depending on a vector of investments $t \in T$ related to a set of available resources from the system. This model recognizes the possibility of having some kind of control over the uncertainty. The aim of these investments is to reduce the existing uncertainty about the performance of the implemented solutions or improve the performance of the system. In order to avoid undesirable effects, some rationality principles have been assumed. We have developed compact formulations for the model under some particular structures for the S_t sets. We have also proposes an approximate method, due to the computational difficulty observed when using the proposed formulations. This method depends on the generation of a sample of feasible investments. The computational experiment has shown that this approximate method could be used in real applications; and possible future research line could be done in the study of how these approximations could be applied in the design of algorithms to solve new robust combinatorial optimization problems. Furthermore, another possible line could be the study of how to extract information from the data and the structure of the problem to generate a better sample of investments. We have also found a bound on the error to asses the quality of the approximate solution, extending in this way previous constant factor approximation results for minmax regret problems without investments.

In Chapter 5, we have presented portfolio problems in which the transaction costs have been turned into decision variables and hence, the financial institution, setting these costs, has become a decision-maker in the problem. We have assumed that the decisions are taken sequentially and we have modeled it using bilevel optimization. We have studied the model in which the financial institution sets the prices first, and then the investor chooses her portfolio trying to maximize de CVaR, ensuring a given expected profit. For this model we have proposed two different MILP formulations. We have also studied the opposite model in which the investor decides first; and we have developed for this one an LP formulation for the case in which no additional constraints are imposed on the prices, and an algorithm for the more general case. Finally, we have addressed the maximum social welfare model. For this last model, we have proposed a MILP formulation and an algorithm based on a Benders decomposition. We have observed in our computational experiment that the bank-leader model seems to return better solutions for both decision-makers, and the social welfare model, as theoretically proved, returns higher aggregated benefits. By making variations in the parameters to model the CVaR and the prices, different bank and investor profiles can be considered. Future research lines could be to consider other

risk measures, to assume continuous ranges of values for the transaction costs or also to incorporate ordered costs in the pricing.

In Chapter 6, we have considered bilevel models for the location of controversial facilities. Primary and secondary facilities had to be located sequentially, in a first and second level, respectively. The first ones as far as possible to the second ones, and the second ones as close as possible to the primary ones. We have developed different MILP formulations and algorithms based on Benders decomposition for the model for different representations of the norm. And we have illustrated them in the ℓ_1 and ℓ_∞ norm cases. Benders algorithms have shown to perform better than the formulations in the computational experiment, being able to solve instances in dimension 20 and with 10000 potential locations for the primary facilities in less than 100 seconds for the ℓ_1 norm case. A possible future research line could be to consider an assignment problem in the lower level or to consider different objective functions for the leader and the follower.

References

Abeledo, H., Fukasawa, R., Pessoa, A., and Uchoa, E. (2013). The time dependent traveling salesman problem: polyhedra and algorithm. *Mathematical Programming Computation*, 5(1):27–55.

Ahuja, R. K., Orlin, J. B., Pallottino, S., and Scutella, M. G. (2003). Dynamic shortest paths minimizing travel times and costs. *Networks: An International Journal*, 41(4):197–205.

Aissi, H., Bazgan, C., and Vanderpooten, D. (2005a). Complexity of the min–max and min–max regret assignment problems. *Operations research letters*, 33(6):634–640.

Aissi, H., Bazgan, C., and Vanderpooten, D. (2005b). Complexity of the min-max (regret) versions of cut problems. In *International Symposium on Algorithms and Computation*, pages 789–798. Springer.

Aissi, H., Bazgan, C., and Vanderpooten, D. (2006). Approximating min-max (regret) versions of some polynomial problems. In *International Computing and Combinatorics Conference*, pages 428–438. Springer.

Aissi, H., Bazgan, C., and Vanderpooten, D. (2007). Approximation of min–max and min–max regret versions of some combinatorial optimization problems. *European Journal of Operational Research*, 179(2):281–290.

Aissi, H., Bazgan, C., and Vanderpooten, D. (2009). Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427–438.

Aksen, D. and Aras, N. (2012). A bilevel fixed charge location model for facilities under imminent attack. *Computers & Operations Research*, 39(7):1364–1381.

Albareda-Sambola, M., Fernández, E., Hinojosa, Y., and Puerto, J. (2009). The multi-period incremental service facility location problem. *Computers & Operations Research*, 36(5):1356–1375.

Albareda-Sambola, M., Hinojosa, Y., and Puerto, J. (2015). The reliable p-median problem with at-facility service. *European Journal of Operational Research*, 245(3):656–666.

Altın, A., Amaldi, E., Belotti, P., and Pınar, M. . (2007). Provisioning virtual private networks under traffic uncertainty. *Networks: An International Journal*, 49(1):100–115.

Angelelli, E., Mansini, R., and Speranza, M. G. (2012). Kernel search: A new heuristic framework for portfolio selection. *Computational Optimization and Applications*, 51(1):345–361.

Aron, I. and Van Hentenryck, P. (2002). A constraint satisfaction approach to the robust spanning tree problem with interval data. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 18–25. Morgan Kaufmann Publishers Inc.

Aronson, J. E. (1989). A survey of dynamic network flows. *Annals of Operations Research*, 20(1):1–66.

Arroyo, J. M. (2010). Bilevel programming applied to power system vulnerability analysis under multiple contingencies. *IET generation, transmission & distribution*, 4(2):178–190.

Atamtürk, A. (2002). On capacitated network design cut-set polyhedra. *Mathematical Programming, Series B*, 92(3):425–437.

Atamtürk, A. and Rajan, D. (2002). On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Mathematical Programming, Series B*, 92(2):315–333.

Averbakh, I. and Berman, O. (2003). An improved algorithm for the minmax regret median problem on a tree. *Networks: An International Journal*, 41(2):97–103.

Averbakh, I. and Lebedev, V. (2004). Interval data min-max regret network optimization problems. *Discrete Applied Mathematics*, 138(3):289–301.

Barahona, F. (1996). Network design using cut inequalities. *SIAM Journal on Optimization*, 6(3):823–837.

Bard, J. F. (2013). *Practical bilevel optimization: algorithms and applications*, volume 30. Springer Science & Business Media.

Baule, R. (2010). Optimal portfolio selection for the small investor considering risk and transaction costs. *OR spectrum*, 32(1):61–76.

Baumann, P. and Trautmann, N. (2013). Portfolio-optimization models for small investors. *Mathematical Methods of Operations Research*, 77(3):345–356.

Bélanger, V., Ruiz, A., and Soriano, P. (2019). Recent optimization models and trends in location, relocation, and dispatching of emergency medical vehicles. *European Journal of Operational Research*, 272(1):1–23.

Ben-Tal, A., El Ghaoui, L., and Nemirovski, A. (2009). *Robust optimization*, volume 28. Princeton University Press.

Benati, S. (2003). The optimal portfolio problem with coherent risk measure constraints. *European Journal of Operational Research*, 150(3):572–584.

Benati, S. (2015). Using medians in portfolio optimization. *Journal of the Operational Research Society*, 66(5):720–731.

Berman, O., Drezner, Z., and Krass, D. (2010). Generalized coverage: New developments in covering location models. *Computers & Operations Research*, 37(10):1675–1687.

Bialas, W. and Karwan, M. (1982). On two-level optimization. *IEEE transactions on automatic control*, 27(1):211–214.

Bigras, L.-P., Gamache, M., and Savard, G. (2008). The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. *Discrete Optimization*, 5(4):685–699.

Blanco, V., Puerto, J., and El Haj Ben Ali, S. (2014). Revisiting several problems and algorithms in continuous location with ℓ_τ- norms. *Computational Optimization and Applications*, 58(3):563–595.

Boonmee, C., Arimura, M., and Asada, T. (2017). Facility location optimization model for emergency humanitarian logistics. *International Journal of Disaster Risk Reduction*, 24:485–498.

Borndörfer, R. and Schlechte, T. (2007). *05. Models for Railway Track Allocation*, volume 7. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Bracken, J. and McGill, J. (1978). Production and marketing decisions with multiple objectives in a competitive environment. *Journal of Optimization Theory and Applications*, 24(3):449–458.

Bracken, J. and McGill, J. T. (1973). Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44.

Bracken, J. and McGill, J. T. (1974). Defense applications of mathematical programs with optimization problems in the constraints. *Operations Research*, 22(5):1086–1096.

Brimberg, J. and Juel, H. (1998). A bicriteria model for locating a semi-desirable facility in the plane. *European journal of operational research*, 106(1):144–151.

Buchheim, C., Liers, F., and Sanità, L. (2011). An exact algorithm for robust network design. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6701 LNCS:7–17.

Cacchiani, V., Caprara, A., and Toth, P. (2010). Scheduling extra freight trains on railway networks. *Transportation Research Part B: Methodological*, 44(2):215–231.

Cacchiani, V. and Toth, P. (2012). Nominal and robust train timetabling problems. *European Journal of Operational Research*, 219(3):727–737.

Calvete, H. I., Galé, C., and Oliveros, M.-J. (2011). Bilevel model for production–distribution planning solved by using ant colony optimization. *Computers & operations research*, 38(1):320–327.

Camacho-Vallejo, J.-F., Cordero-Franco, Á. E., and González-Ramírez, R. G. (2014). Solving the bilevel facility location problem under preferences by a stackelberg-evolutionary algorithm. *Mathematical Problems in Engineering*, 2014.

Candler, W. and Norton, R. (1977). Multilevel programing. *Technical Report*, 20.

Castro, F., Gago, J., Hartillo, I., Puerto, J., and Ucha, J. (2011). An algebraic approach to integer portfolio problems. *European Journal of Operational Research*, 210(3):647–659.

Catanzaro, D., Labbé, M., and Salazar-Neumann, M. (2011). Reduction approaches for robust shortest path problems. *Computers & operations research*, 38(11):1610–1619.

Chassein, A. and Goerigk, M. (2015). A new bound for the midpoint solution in min-max regret optimization with an application to the robust shortest path problem. *European Journal of Operational Research*, 244(3):739–747.

Cherkassky, B., Goldberg, A., and Radzik, T. (1996). Shortest paths algorithms: theory and experimental evaluation. *Mathematical Programming*, pages 129–174.

Church, R. L. and Scaparra, M. P. (2007). Protecting critical assets: the r-interdiction median problem with fortification. *Geographical Analysis*, 39(2):129–146.

Colson, B., Marcotte, P., and Savard, G. (2005). Bilevel programming: A survey. *4or*, 3(2):87–107.

Conde, E. (2004). An improved algorithm for selecting p items with uncertain returns according to the minmax-regret criterion. *Mathematical Programming*, 100(2):345–353.

Conde, E. (2007). Minmax regret location–allocation problem on a network under uncertainty. *European Journal of Operational Research*, 179(3):1025–1039.

Conde, E. (2010). A 2-approximation for minmax regret problems via a mid-point scenario optimal solution. *Operations Research Letters*, 38(4):326–327.

Conde, E. (2012). On a constant factor approximation for minmax regret problems using a symmetry point scenario. *European Journal of Operational Research*, 219(2):452–457.

Conde, E. and Leal, M. (2017). Minmax regret combinatorial optimization problems with investments. *Computers & Operations Research*, 85:1–11.

Conde, E. and Leal, M. (2019). A scenario based optimization model for the design of a robust distribution network. Technical report, Universidad de Sevilla.

Conde, E., Leal, M., and Puerto, J. (2018). A minmax regret version of the time-dependent shortest path problem. *European Journal of Operational Research*, 270(3):968–981.

Cooke, K. L. and Halsey, E. (1966). The shortest route through a network with time-dependent internodal transit times. *Journal of mathematical analysis and applications*, 14(3):493–498.

Costa, A. (2005). A survey on benders decomposition applied to fixed-charge network design problems. *Computers and Operations Research*, 32(6):1429–1450.

Dempe, S. (2002a). *Foundations of bilevel programming*. Springer Science & Business Media.

Dempe, S. (2002b). Foundations of bilevel programming. nonconvex optimization and its applications. *61 (2002)*, 61.

Dempe, S., Kalashnikov, V., Pérez-Valdés, G. A., and Kalashnykova, N. (2015). Bilevel programming problems. *Energy Systems. Springer, Berlin.*

DeNegre, S. T. and Ralphs, T. K. (2009). A branch-and-cut algorithm for integer bilevel linear programs. In *Operations research and cyber-infrastructure*, pages 65–78. Springer.

Erkut, E. and Neuman, S. (1989). Analytical models for locating undesirable facilities. *European Journal of Operational Research*, 40(3):275–291.

Fahimnia, B., Farahani, R., Marian, R., and Luong, L. (2013). A review and critique on integrated production–distribution planning models and techniques. *Journal of Manufacturing Systems*, 32(1):1–19.

Fereiduni, M. and Shahanaghi, K. (2017). A robust optimization model for distribution and evacuation in the disaster response phase. *Journal of Industrial Engineering International*, 13(1):117–141.

Fernández, E., Pozo, M., and Puerto, J. (2014). Ordered weighted average combinatorial optimization: Formulations and their properties. *Discrete Applied Mathematics*, pages 97–118.

Fischer, F. and Helmberg, C. (2014). Dynamic graph generation for the shortest path problem in time expanded networks. *Mathematical Programming*, 143(1-2):257–297.

Fischetti, M., Ljubić, I., Monaci, M., and Sinnl, M. (2017). A new general-purpose algorithm for mixed-integer bilevel linear programs. *Operations Research*, 65(6):1615–1637.

Fischetti, M., Ljubić, I., Monaci, M., and Sinnl, M. (2018). On the use of intersection cuts for bilevel optimization. *Mathematical Programming*, 172(1-2):77–103.

Fortz, B., Labbé, M., Louveaux, F., and Poss, M. (2013). Stochastic binary problems with simple penalties for capacity constraints violations. *Mathematical programming*, 138(1-2):199–221.

Furini, F., Persiani, C. A., and Toth, P. (2016). The time dependent traveling salesman planning problem in controlled airspace. *Transportation Research Part B: Methodological*, 90:38–55.

Garey, M. R. (1979). A guide to the theory of np-completeness. *Computers and intractability.*

Gawiejnowicz, S. (2008). *Time-dependent scheduling.* Springer Science & Business Media.

Günlük, O. (1999). A branch-and-cut algorithm for capacitated network design problems. *Mathematical Programming, Series B*, 86(1):17–39.

Griffith, R. L., Strowe, R. J., Newell, J. C., Edic, P. M., Messina, R. F., and Houghton, F. C. (1998). Bi-level charge pulse apparatus to facilitate nerve location during peripheral nerve block procedures. US Patent 5,853,373.

Gutiérrez, G., Kouvelis, P., and Kurawarwala, A. (1996). A robustness approach to uncapacitated network design problems. *European Journal of Operational Research*, 94(2):362–376.

Hammad, A. W., Rey, D., and Akbarnezhad, A. (2018). A bi-level mixed integer programming model to solve the multi-servicing facility location problem, minimising negative impacts due to an existing semi-obnoxious facility. In *Data and Decision Sciences in Action*, pages 381–395. Springer.

Hansen, P., Jaumard, B., and Savard, G. (1992). New branch-and-bound rules for linear bilevel programming. *SIAM Journal on scientific and Statistical Computing*, 13(5):1194–1217.

Hashemi, S. M., Mokarami, S., and Nasrabadi, E. (2010). Dynamic shortest path problems with time-varying costs. *Optimization Letters*, 4(1):147–156.

He, X., Sun, X., and Von Laszewski, G. (2003). Qos guided min-min heuristic for grid task scheduling. *Journal of Computer Science and Technology*, 18(4):442–451.

Hendricks, K. and Singhal, V. (2005). An empirical analysis of the effect of supply chain disruptions on long-run stock price performance and equity risk of the firm. *Production and Operations Management*, 14(1):35–52.

Heyman, D. P. and Sobel, M. J. (2004). *Stochastic models in operations research: stochastic optimization*, volume 2. Courier Corporation.

Jabbarzadeh, A., Fahimnia, B., and Sheu, J.-B. (2017). An enhanced robustness approach for managing supply and demand uncertainties. *International Journal of Production Economics*, 183:620–631.

Jeroslow, R. G. (1985). The polynomial hierarchy and a simple model for competitive analysis. *Mathematical programming*, 32(2):146–164.

Kalcsics, J., Nickel, S., Pozo, M. A., Puerto, J., and Rodríguez-Chía, A. M. (2014). The multicriteria p-facility median location problem on networks. *European Journal of Operational Research*, 235(3):484–493.

Karasan, O. E., Pinar, M. e., and Yamana, H. (2001). The robust shortest path problem with interval data. *Optimization Online*.

Kasperski, A. (2008). *Discrete optimization with interval data. Minmax regret and fuzzy approach*. Studies in Fuzziness and Soft Computing. Springer.

Kasperski, A. and Zielinski, P. (2006). An approximation algorithm for interval data minmax regret combinatorial optimization problems. *Inf. Process. Lett.*, 97(5):177–180.

Kellerer, H., Mansini, R., and Speranza, M. G. (2000). Selecting portfolios with fixed costs and minimum transaction lots. *Annals of Operations Research*, 99(1-4):287–304.

Klinkowski, M., Herrero, F., Careglio, D., and Solé-Pareta, J. (2005). Adaptive routing algorithms for optical packet switching networks. *In Optical Network Design and Modeling.*, 49(1):235–241.

Kolm, P. N., Tütüncü, R., and Fabozzi, F. J. (2014). 60 years of portfolio optimization: Practical challenges and current trends. *European Journal of Operational Research*, 234(2):356–371.

Konno, H., Akishino, K., and Yamamoto, R. (2005). Optimization of a long-short portfolio under nonconvex transaction cost. *Computational Optimization and Applications*, 32(1-2):115–132.

Kouvelis, P. and Yu, G. (1997). *Robust Discrete Optimization and Its Applications*. Kluwer Academic Publisher.

Krejić, N., Kumaresan, M., and Rožnjik, A. (2011). Var optimal portfolio with transaction costs. *Applied Mathematics and Computation*, 218(8):4626–4637.

Labbé, M., Leal, M., and Puerto, J. (2018). New models for the location of controversial facilities: A bilevel programming approach. Technical report, Université Libre de Bruxelles- Universidad de Sevilla. `https://hal.inria.fr/hal-01933601`.

Labbé, M., Peeters, D., and Thisse, J.-F. (1995). Location on networks. *Handbooks in operations research and management science*, 8:551–624.

Labbé, M. and Violin, A. (2013). Bilevel programming and price setting problems. *4OR*, 11(1):1–30.

Le Thi, H. A., Moeini, M., and Dinh, T. P. (2009). Dc programming approach for portfolio optimization under step increasing transaction costs. *Optimization*, 58(3):267–289.

Leal, M., Ponce, D., and Puerto, J. (2018). An extended version of portfolio problems with two levels decision-makers: Optimal portfolio selection with pricing decisions on transaction costs. Technical report, arXiv preprint arXiv:1804.04174.

Lee, C., Lee, K., and Park, S. (2013). Benders decomposition approach for the robust network design problem with flow bifurcations. *Networks*, 62(1):1–16.

Loridan, P. and Morgan, J. (1996). Weak via strong stackelberg problem: new results. *Journal of global Optimization*, 8(3):263–287.

Love, R. F., Morris, J. G., and Wesolowsky, G. O. (1988). Facilities location. *Chapter*, 3:51–60.

Luenberger, D. G., Ye, Y., et al. (1984). *Linear and nonlinear programming*, volume 2. Springer.

Ma, S. (2016). A nonlinear bi-level programming approach for product portfolio management. *SpringerPlus*, 5(1):727.

Ma, T., Yan, Q., Liu, W., Guan, D., and Lee, S. (2011a). Grid task scheduling: algorithm review. *IETE Technical Review*, 28(2):158–167.

Ma, T., Yan, Q., Liu, W., and Mengmeng, C. (2011b). A survey on grid task scheduling. *International Journal of Computer Applications in Technology*, 41(3-4):303–309.

Magnanti, T. and Wong, R. (1984). Network design and transportation palnning: Models and algorithms. *Transportation Science*, 18(1):1–55.

Mansini, R., Ogryczak, W., and Speranza, M. G. (2003). On lp solvable models for portfolio selection. *Informatica*, 14(1):37–62.

Mansini, R., Ogryczak, W., and Speranza, M. G. (2014). Twenty years of linear programming based portfolio optimization. *European Journal of Operational Research*, 234(2):518–535.

Mansini, R., Ogryczak, W., and Speranza, M. G. (2015). *Quantitative Financial Risk Management: Theory and Practice*. (eds. Zopounidis, Constantin and Galariotis, Emilios), John Wiley & Sons.

Mansini, R. and Speranza, M. G. (2005). An exact approach for portfolio selection with transaction costs and rounds. *IIE transactions*, 37(10):919–929.

Marcotte, P. (1986). Network design problem with congestion effects: A case of bilevel programming. *Mathematical programming*, 34(2):142–162.

Marín, A., Martínez-Merino, L. I., Rodríguez-Chía, A. M., and Saldanha-da Gama, F. (2018). Multi-period stochastic covering location problems: Modeling framework and solution approach. *European Journal of Operational Research*, 268(2):432–449.

Markowitz, H. (1952). Portfolio selection. *The journal of finance*, 7(1):77–91.

McCormick, G. P. (1976). Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems. *Mathematical programming*, 10(1):147–175.

Melachrinoudis, E. and Xanthopulos, Z. (2003). Semi-obnoxious single facility location in euclidean space. *Computers & Operations Research*, 30(14):2191–2209.

Merton, R. C. (1969). Lifetime portfolio selection under uncertainty: The continuous-time case. *The review of Economics and Statistics*, pages 247–257.

Minoux, M. (2010). Robust network optimization under polyhedral demand uncertainty is np-hard. *Discrete Applied Mathematics*, 158(5):597–603.

Montemanni, R. and Gambardella, L. M. (2004). An exact algorithm for the robust shortest path problem with interval data. *Computers & Operations Research*, 31(10):1667–1680.

Montemanni, R. and Gambardella, L. M. (2005a). A branch and bound algorithm for the robust spanning tree problem with interval data. *European Journal of Operational Research*, 161(3):771–779.

Montemanni, R. and Gambardella, L. M. (2005b). The robust shortest path problem with interval data via benders decomposition. *4or*, 3(4):315–328.

Montemanni, R., Gambardella, L. M., and Donati, A. V. (2004). A branch and bound algorithm for the robust shortest path problem with interval data. *Operations Research Letters*, 32(3):225–232.

Moore, J. T. and Bard, J. F. (1990). The mixed integer linear bilevel programming problem. *Operations research*, 38(5):911–921.

Nickel, S. and Puerto, J. (2006). *Location theory: a unified approach*. Springer Science & Business Media.

Nickel, S., Saldanha-da Gama, F., and Ziegler, H.-P. (2012). A multi-stage stochastic supply network design problem with financial decisions and risk management. *Omega*, 40(5):511–524.

Olivares-Nadal, A. V. and DeMiguel, V. (2018). A robust perspective on transaction costs in portfolio optimization. *Operations Research*.

Owen, S. H. and Daskin, M. S. (1998). Strategic facility location: A review. *European journal of operational research*, 111(3):423–447.

Pan, F. and Nagi, R. (2010). Robust supply chain design under uncertain demand in agile manufacturing. *Computers and Operations Research*, 37(4):668–683.

Peng, P., Snyder, L., Lim, A., and Liu, Z. (2011). Reliable logistics networks design with facility disruptions. *Transportation Research Part B: Methodological*, 45(8):1190–1211.

Pereira, J. and Averbakh, I. (2013). The robust set covering problem with interval data. *Annals of Operations Research*, 207(1):217–235.

Pessoa, A., Uchoa, E., de Aragão, M. P., and Rodrigues, R. (2010). Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation*, 2(3-4):259–290.

Picard, J.-C. and Queyranne, M. (1978). The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, 26(1):86–110.

Puerto, J., Ricca, F., and Scozzari, A. (2011). Minimax regret path location on trees. *Networks*, 58(2):147–158.

Puerto, J., Rodríguez-Chía, A. M., and Tamir, A. (2009). Minimax regret single-facility ordered median location problems on networks. *INFORMS Journal on Computing*, 21(1):77–87.

Puerto, J., Rodríguez-Chía, A. M., and Tamir, A. (2017). Revisiting k-sum optimization. *Mathematical Programming*, 165(2):579–604.

Rockafellar, R. T. (1970). *Convex analysis*. Princeton university press.

Rockafellar, R. T., Uryasev, S., et al. (2000). Optimization of conditional value-at-risk. *Journal of risk*, 2:21–42.

Rosenhead, J., Elton, M., and Gupta, S. (1972). Robustness and optimality as criteria for strategic decisions. *Operational Research Quaterly*, 23(4):413–430.

Scaparra, M. P. and Church, R. L. (2008). A bilevel mixed-integer program for critical infrastructure protection planning. *Computers & Operations Research*, 35(6):1905–1923.

Schmid, V. and Doerner, K. F. (2010). Ambulance location and relocation problems with time-dependent travel times. *European journal of operational research*, 207(3):1293–1303.

Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2009). *Lectures on stochastic programming: modeling and theory*. SIAM.

Stackelberg, H. v. et al. (1952). Theory of the market economy.

Takahashi, R., Ramirez, J., Vasconcelos, J., and Saldanha, R. (2001). Sensitivity analysis for optimization problems solved by stochastic methods. *IEEE transactions on magnetics*, 37(5):3566–3569.

Taş, D., Gendreau, M., Jabali, O., and Laporte, G. (2016). The traveling salesman problem with time-dependent service times. *European Journal of Operational Research*, 248(2):372–383.

Tijms, H. C. and Tijms, H. C. (1994). *Stochastic models: an algorithmic approach*, volume 994. John Wiley & Sons Chichester.

Valle, C., Meade, N., and Beasley, J. (2014). Absolute return portfolios. *Omega*, 45:20–41.

Vander Wiel, R. J. and Sahinidis, N. V. (1996). An exact solution approach for the time-dependent traveling-salesman problem. *Naval Research Logistics (NRL)*, 43(6):797–820.

Vicente, L., Savard, G., and Júdice, J. (1994). Descent approaches for quadratic bilevel programming. *Journal of Optimization Theory and Applications*, 81(2):379–399.

Vicente, L. N. and Calamai, P. H. (1994). Bilevel and multilevel programming: A bibliography review. *Journal of Global optimization*, 5(3):291–306.

Woodside-Oriakhi, M., Lucas, C., and Beasley, J. E. (2013). Portfolio rebalancing with an investment horizon and transaction costs. *Omega*, 41(2):406–420.

Yaman, H., KaraşAn, O. E., and PıNar, M. Ç. (2001). The robust spanning tree problem with interval data. *Operations research letters*, 29(1):31–40.

Ye, Y. (2004). Linear conic programming. *Manuscript. Stanford University, Stanford, CA*.

Yu, G. and Yang, J. (1998). On the robust shortest path problem. *Computers & Operations Research*, 25(6):457–468.