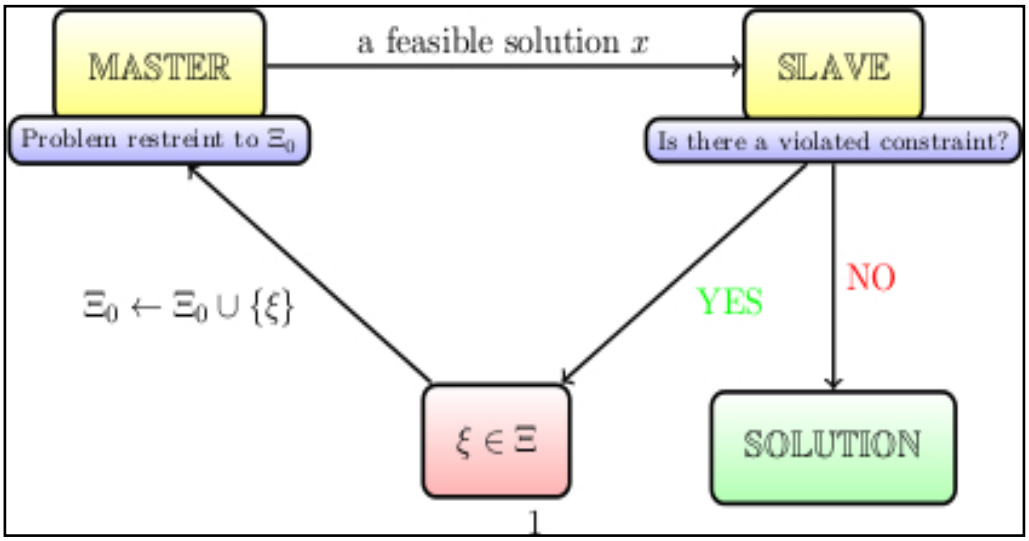
Contributions to static and adjustable robust linear optimization

Marcio Costa Santos

Par **Marcio COSTA SANTOS**

Contributions to static and adjustable robust linear optimization

Thèse présentée
pour l'obtention du grade
de Docteur de l'UTC



Soutenue le 25 novembre 2016
Spécialité : Technologies de l'Information et des Systèmes :
Unité de recherche Heudyasic (UMR-7253)
D2312

Université de Technologie de Compiègne

École Doctorale

Doctorat

Technologies de l'Information et des Systèmes

Marcio Costa Santos

Contributions to static and adjustable robust linear optimization

Soutenue le 25 novembre 2016

Jury :

Agostinho Agra (examinateur): professeur, Universidade de Aveiro.
Jacques Carlier (examinateur): professeur émérite, Université de Technologie de Compiègne.
Marie-Christine Costa (rapporteur): professeur, ENSTA.
Dritan Nace (co-directeur de thèse): professeur, Université de Technologie de Compiègne.
Artur Alves Pessoa (rapporteur): professeur, Universidade Federal Fluminense.
Michael Poss (co-directeur de thèse): chargé de recherche, Université de Montpelier.

UMR CNRS 7253 Heudiasyc, Université
de Technologie de Compiègne,
Centre de Recherches de Royallieu
Compiègne, France

2016

We cannot solve our problems with the same thinking we used when we created them.
Albert Einstein

Acknowledgments

First, I would like to thank both my advisors, Michael Poss and Dritan Nace, for their support, valuable guidance (not restricted to the topic of this thesis) and opportunity provided to me. As well as the all the people from Heudiasyc and UTC for the welcome.

Second, I want to express my gratitude towards all the people that helped me to became a better research and professional. Agostinho Agra for his patience and mathematical rigor that many times changed the path of this work, Manoel Campelo for his work ethic and availability for help with doubts, many times sillies, and my friends from ParGO and Heudiasyc for the fruitful discussions.

Third, I would like to thank all the people that made the days of hard work a little bit less hard. Wendell Diniz, Leonardo Trigueiro, Sagnik Datta, Giovanni Bernadi, Ana Maria, Marcio Fuckner and all the guys from Compiegne, for their companionship, beers, and many not so fruitful discussions. Elvys Linhares, Pedro Henrique, Israel Zinc and all the guys from Avignon, for all the funny days as well as the long nights. Livia Almada, Ticiana Linhares, Tatiane Fernandes, Arthur Araruna, Julio Araujo and all my friends in the ParGO team for the cozy welcome every time I went back to Brazil.

Last but not least, I would like to say a special thanks to all my family, especially my brother, Marcelo Costa Santos, that encouraged me to follow this path. A special thanks is also needed to my friends Ana Karolina Maia and Camila Nair Aguiar that were very supportive even when they do not agree with me, and more worried about me and my future than myself.

Resumé

L'incertitude a été toujours présente dans les problèmes d'optimisation. Dans ce travail, nous nous intéressons aux problèmes d'optimisation multi-niveaux où l'incertitude apparaît très naturellement. Les problèmes d'optimisation multi-niveaux avec incertitude ont suscité un intérêt à la fois théorique et pratique. L'optimisation robuste fait partie des méthodes les plus étudiées pour traiter ces problèmes.

En optimisation robuste, nous cherchons une solution qui optimise la fonction objective pour le pire scénario appartenant à une ensemble d'incertitude donné. Les problèmes d'optimisation robuste multi-niveaux sont difficiles à résoudre, même de façon heuristique. Dans cette thèse, nous abordons les problèmes d'optimisation robuste à travers le prisme des méthodes de décomposition. Ces méthodes décomposent le problème en un problème maître (MP) et plusieurs problèmes satellites de séparation (AP). Le problème maître contient les contraintes robustes initiales, mais écrites uniquement pour un nombre fini de scénarios. D'autres scénarios sont générés au fur et à mesure par la résolution des problèmes satellites de séparation.

Dans ce contexte, les solutions et les relaxations heuristiques ont une importance particulière. Même pour les problèmes d'optimisation combinatoires, les relaxations sont importantes pour analyser l'écart de l'optimalité des solutions heuristiques. Un autre aspect important est l'utilisation des heuristiques comme integrés dans une méthode exacte. Ansi, les solutions heuristiques permettent un gain important en terme de temps de calcul, entre autres, parce que les problèmes satellites de séparation doivent être résolus à plusieurs reprises. Une bonne solution heuristique pour les problèmes satellites de séparation peut nous permettre de générer un scénario supplémentaire pour le problème maître.

Les principales contributions de ce travail sont les suivantes. Premièrement, nous proposons une nouvelle relaxation pour les problèmes multi-niveaux basée sur l'approche dite

d'information parfaite dans le domaine de l'optimisation stochastique. L'idée principale derrière cette méthode est d'éliminer les contraintes de non anticipativité du modèle pour obtenir un problème plus simple. Nous pouvons ensuite fournir des algorithmes combinatoires ad-hoc et des formulations de programmation mixte en nombres entiers compactes pour ce problème. Deuxièmement, nous proposons de nouveaux algorithmes de programmation dynamique pour résoudre les problèmes satellites apparaissant dans une classe spécifique de problèmes robustes pour un ensemble d'incertitude de type budget. Ce type d'incertitude est basé sur le nombre maximum d'écarts autorisés et leur taille. Ces algorithmes peuvent être appliqués à des problèmes de lot-sizing et à des problèmes de tournées de véhicules. Enfin, nous proposons un modèle robuste pour un problème lié à l'installation équitable de capteurs. Ce modèle fait le lien entre l'optimisation robuste et l'optimisation stochastique avec contraintes probabilistes ambigües.

Mots-clés: optimisation robuste, optimisation sous incertitude, complexité computationnelle, problèmes de lot-sizing, problèmes à plusieurs étapes.

Abstract

Uncertainty has always been present in optimization problems, and it arises even more severely in multistage optimization problems. Multistage optimization problems under uncertainty have attracted interest from both the theoretical and the practical level. Robust optimization stands among the most established methodologies for dealing with such problems.

In robust optimization, we look for a solution that optimizes the objective function for the worst possible scenario, in a given uncertainty set. Robust multi-stage optimization problems are hard to solve even heuristically. In this thesis, we address robust optimization problems through the lens of decompositions methods. These methods are based on the decomposition of the robust problem into a master problem (MP) and several adversarial separation problems (APs). The master problem contains the original robust constraints, however, written only for finite numbers of scenarios. Additional scenarios are generated on the fly by solving the APs.

In this context, heuristic solutions and relaxations have a particular importance. Similarly to combinatorial optimization problems, relaxations are important to analyze the optimality gap of heuristic solutions. Heuristic solutions represent a substantial gain from the computational viewpoint, especially when used to solve the separation problem. Because the adversarial problems must be solved several times, good heuristic solution may avoid the exact solution of the APs.

The main contributions of this work are three-fold. First, we propose a new relaxation for multi-stage problems based on the approach named *perfect information* in the field of stochastic optimization. The main idea behind this method is to remove non-anticipativity constraints from the model to obtain a simpler problem for which we can provide ad-hoc combinatorial algorithms and compact mixed integer programming formulations. Second, we propose new dynamic programming algorithms to solve the APs

for robust problems involving budgeted uncertainty, which are based on the maximum number of deviations allowed and on the size of the deviations. These algorithms can be applied to lot-sizing problems and vehicle routing problems among others. Finally, we study the robust equitable sensor location problem. We make the connection between the robust optimization and the stochastic programming with ambiguous probabilistic constraints. We propose linear models for several variants of the problem together with numerical results.

Keywords: robust optimization, optimization over uncertainty, computational complexity, lot-sizing problems, multi-stage problems.

List of Figures

List of Tables

Contents

Introduction

Uncertainty has always been present in optimization problems. This is due, among others, to the possibility of data errors and the lack of information about some parameter of the problem at the moment we solve it. Such uncertainty arises naturally in problems where the decisions can be divided into disjoint sets associated with periods of time and the taken decisions at a period are influenced by the decisions taken at earlier periods. These problems are named multi-stage problems, because of the temporal relation between the decision. They will be one of the focus of our work.

Multistage optimization problems appear in insurance [MS04], energy production [HCF10] and trading [VJM16], logistics and transportation [BMdO15], among other areas. Multistage optimization problems under uncertainty have attracted interest from both, theoretical and practical points-of-view from diverse research communities as we can see in [LF14], [Tim15], [PP15] and [AP14]. Multi-stage optimization problems are hard to model properly due to the links between the uncertain parameter revealed in early periods and the decisions that must be taken in the future. More precisely, we must enforce that for any two different scenarios (a concrete realization of the uncertain parameters is called a *scenario*), that are equals until the period i, we must have the same decisions for the decision variables until period i. Such kind of constraints, called *non-anticipativity constraints* model the intuition that the decision maker can only take into consideration something that already happens and these constraints usually need infinite numbers of variables and linear constraints to be modeled accurately. Among the most established methodologies for dealing with such problems are stochastic programming and, more

recently, robust optimization.

In robust optimization [BTEGN09], we look for a solution that optimizes the objective function according to "worst possible" realization of the uncertain parameters. Thanks to such a strong constraint, robust optimization models do not make any assumption about the probabilistic distribution of the uncertain parameters. It rather assumes that all possible scenarios lie into a given uncertainty set, which is usually considered to be a tractable convex set [BN98]. Notice that the uncertainty set can be seen as the set of scenarios against which we want to be protected. Robust optimization often conducts to reasonable solutions that are practical, and it makes only a few simple assumptions about the uncertain parameters of the problem [BGGN04], [BS03]. Robust solutions are usually more expensive, from the point of view of the evaluation of the objective function, than stochastic ones. In this work, we focus on robust optimization solutions and robust programming methods.

Some approaches to solving robust problems are based on decomposition methods, which are especially useful for multi-stage problems that only have two-stages or does not have non-anticipativity constraints. These methods are based on the idea that to solve a robust optimization problem we can decompose the problem into a master problem (MP) and several adversarial separation problems (APs). The master problem contains the original robust constraints, however, written only for a finite number of scenarios. Additional scenarios are generated on the fly by solving the APs. In this framework, the master problem provides optimality while the adversarial problems check feasibility.

In this context, heuristic solutions and relaxations have a special importance. Heuristic solutions are feasible solutions that may be not optimal. One example is the *affine decision rules* presented in [BIP10], which considers that the optimization variables are affine functions of the uncertain parameters. Relaxations, as the *dual affine decision rules* presented in [KWG11], are important to analyze the *optimally gap* of heuristic solutions defined as the difference between the relaxation and a heuristic solution. This gap represents in some way the quality of the proposed solution. Hence good relaxations

(relaxations which have the value close to the optimal solution) are important for a good estimation of the quality of a heuristic solution.

With respect to decompositions methods for robust optimization, heuristic solutions may represent a substantial gain from the computational point-of-view when this is used to solve the AP problems. We recall that such problems need to be solved several times, not necessary to optimality. A good heuristic solution may prove the non-feasibility of a solution for the master problem and allow us to generate an additional scenario for the master problem.

A kind of robust optimization problems that have been receiving a lot of attention are *robust lot-sizing problems*. Such problems are naturally multi-stage and, despite their simplicity they conserve the main difficulties and structures of general robust multi-stage optimization problems. Special attention is given to these robust optimization problems, because not only such problems can, typically, be solved in a tractable manner but they provide useful insights into the structure of the multi-stage problems in general.

One of the contributions of this work is to propose a new relaxation method based on the famous approach of *perfect information* for stochastic optimization, sometimes also known as the *clairvoyant version*. The main idea behind this method is to remove the non-anticipativity constraints of the model to obtain a simpler problem, which is easier to solve. When we remove the non-anticipativity constraints, each scenario of the uncertainty set has a set of variables associated with it, and the problem becomes a two-stage robust problem. Remark that we can see this problem as the problem where the decision-maker has full knowledge of the future, that is why this method is usually called the clairvoyant version. Because we commonly end up with a robust two-stage problem after removing the non-anticipativity constraints, we also proposed a Benders-like algorithm to solve this kind of problems when it possesses a special dependency structure of the uncertainty and a special structure on the uncertainty set.

This thesis is organized as follows. Chapter 1 presents concepts used in the field of optimization under uncertainty. We present the main methods known in the literature

to deal with uncertainty in optimization problems, namely, stochastic and robust optimization. This work is focused on robust optimization. Hence we present stochastic optimization very briefly at the end of the chapter just to better place the contributions presented. In the second part, we go over classical results and tools for robust optimization, namely, affine decision rules, dualization and decomposition methods. At this point we note that affine decisions rules give heuristic solutions although they might be optimal for several cases [BIP10], they can also provide poor solutions in others.

In Chapter 2, we present the main definitions and concepts in the field of inventory management as well as we formalize the definition of many variants of lot-sizing problems and robust lot-sizing problems. Chapter 2 also presents the most important notations used in the thesis, as well as the concepts of backlog and setup costs.

In Chapter 3, we consider the budgeted uncertainty polytope from Bertsimas and Sim, widely used in the literature, and propose new dynamic programming algorithms to solve the adversarial separation problems which are based on the maximum number of deviations allowed and on the size of such deviations. Our algorithms can be applied to robust constraints that occur in various applications such as lot-sizing, traveling salesperson problems with time-windows, scheduling problems, and inventory routing problems, among many others. We show how the simple version of the algorithms leads to a fully polynomial approximation scheme when the deterministic problem is convex. We assess our approach to a lot-sizing problem numerically, showing a comparison with the classical MIP reformulation of the AP traditionally used in the literature.

In Chapter 4, we address the problem of obtaining relaxations for multi-stage robust problems, with a special focus on robust lot-sizing problems. We define a relaxation problem based on the approach of perfect information, and we study the relaxation problem associated with different versions of the robust lot-sizing problem. Most relaxation problems for the robust lot-sizing problem are proven to be polynomial, when the uncertainty polytope is *simple enough* (this will be defined formally in the chapter, but it can be understood as a compact linear polytope). We also discuss the importance of

such relaxation from the theoretical point-of-view and compare them, numerically, with the relaxation presented in [KWG11], one of the few references concerning relaxation in robust multi-stage optimization.

In Chapter 5, we change a little bit the focus from robust multi-stage problems and robust lot-sizing problems to equitable and ambiguous sensor location problems. We tackle three sensor locations problems: equitable sensor location, resilient sensor location, and ambiguous sensor location. In the equitable sensor location, one wants to determine a placement of K sensors in a plan to protect locations of interest in such way that each location of interest has the highest equitable level of probabilistic protection. In the resilient sensor location problem, we extend the basic equitable model to the case when sensors are subject to failures. In the ambiguous sensor location problem, we consider the case where the surveillance probabilities that an intruder is detected are not known with accuracy. We study these three problems and present models and solution methods for each one of them.

To finalize, Chapter 6 presents a conclusion of all the work done as well as some open questions for future work.

Part I

General Context and Preliminaries

The first part of this thesis is devoted to the description of the context, the research area and main methods and techniques to be used. We start with robust optimization, which is the main focus of this thesis. Next, we recall the main techniques and methods in the field of robust optimization. We present a more formal definition of robust multi-stage problems as well as the popular approaches like these based on affine decision rules, dualization and Bender's decomposition for such problems.

Chapter 1

Uncertainty in Optimization Problems

1.1 Problems with uncertainty

Optimization refers to the analysis and solution of problems characterized by a feasible region, which contains the possible solutions (*feasible solution*) and an objective function that gives an evaluation of each possible solution. The goal is to determine the solution with the largest profit or the smallest cost, depending on the statement of the problem.

Commonly, feasible solutions are represented by a set of variables that can take values in a fixed range, defining the feasible region for the problem. The variables can be seen as decisions that should be taken by someone. These methods maximize (if we want a feasible solution with the largest profit) or minimize (if we want a feasible solution with the smallest cost) the objective function evaluation.

Mathematically, we can write an optimization problem as

$$\begin{aligned} \min \quad (or \max) \quad & c(x) \\ st \quad & f_i(x) \leq b_i \quad \forall i \in \{1, \ldots, n\} \\ & x \in \mathcal{X}, \end{aligned}$$

where x is the vector of *optimization variables*, $\mathcal{X}$ is a set of possible values (choices) for x, $f_i : \mathcal{X} \to \mathbb{R}$ is a function and b_i a real number for each $i \in \{1, \ldots, n\}$, which define the restrictions of the feasible region associated to the problem; and $c : \mathcal{X} \to \mathbb{R}$ is the objective function.

It is commonly known that uncertainties are frequent in the data related to optimization problems. Optimization problems under uncertainty are characterized by the need for the decision maker to take their decisions without knowing what their full effects will be. Such problems appear in many areas and present many exciting challenges in theory and practice [Roc01]. Some examples of applications where one must take into consideration data uncertainty follow:

Generation of electrical power: Electrical utilities must decide their production and distribution without knowing exactly the client demands [GCCP93] and [GKHR03].

Reservoir management: Systems of water reservoirs and their connecting flows have to be managed in a reasonable manner for drinking water, irrigation, hydropower, flood control, recreation, and the protection of fish runs. However, none of these demands are known or even can be guaranteed to exist [RM10].

Portfolio selection: Investments must be made in stocks, bonds, foreign currencies and without knowing for sure how their values will rise or fall [Zhu10].

The uncertainty can be related to errors while measuring the data associated to the problem [GH86]. It can also be intrinsic to the problem, for example when a parameter of the problem models some information that is related to the future or to elements that can

not be objectively measured. This implies that the uncertainty can influence the functions f_i, as well as the real values b_i and the objective function c. Mathematically they become functions of the uncertainty, or to be more precise of the *uncertain parameters* that wrap up the uncertainty related to the problem. We assume that the uncertain parameters belong to a set denoted Ξ which we call *the uncertainty set*.

In this context, one wants to optimize a function g that takes as arguments the value of the function $c()$ for each element $\xi \in \Xi$. More precisely, we can write an optimization problem under uncertainty as follows:

$$\begin{aligned} \min(\max) \quad & g(\{c(x,\xi)|\xi \in \Xi\}) \\ st \quad & f_i(x,\xi) \leq b_i(\xi) \qquad \forall i \in \{1,\ldots,n\}, \quad \forall \xi \in \Xi \\ & x \in \mathcal{X}. \end{aligned}$$

For instance, g can represent the expectation or the maximum value taken over Ξ.

1.2 Models for the uncertain parameters

One crucial point concerns the mathematical model used to represent the uncertain parameters linked to the uncertainty present in the problem. Notice that the model of such uncertain parameters is a challenge in itself, and many research papers have seeked to achieve a meaningful representation of the possible uncertainties present in an optimization problem. Herein, we focus on two kinds of mathematical models for the uncertain parameters: *scenario's uncertainty* and *polyhedral uncertainty*.

In scenario's uncertainty, we have a list of all possible realizations of the uncertain parameters. This method is inherently discrete since such list must be finite [God00], [CTC07]. More precisely, the uncertainty set Ξ shall consist of a list $\{\xi_1, \xi_1, \ldots, \xi_m\}$.

On the other hand, polyhedral uncertainty assumes that each possible realization of the uncertain parameters lies into a polyhedral set that is a closed set defined by a group

of linear constraints [Min09],[AYP11]. More precisely,

$$\Xi = \{ \quad \xi \quad | \quad W\xi \leq q\}$$

where W is a matrix, q is a vector with adequate dimensions.

A specific polyhedral uncertainty set called the *budgeted uncertainty polytope* [BS04] has gained much attention in the last years due to its simplicity. In addition, the budgeted uncertainty polytope often allows us to adapt the algorithms for the problem without uncertainty (called *the nominal version*) to the version with uncertainty. The budgeted uncertainty polytope is defined according to a real number Γ that roughly represents the number of elements in the uncertain parameters that are allowed to vary, more precisely,

$$\Xi_\Gamma = \{ \quad \xi \quad | \quad ||\xi||_1 \leq \Gamma; \quad -1 \leq \xi \leq 1\}.$$

We denote this set by Ξ_Γ in order to indicate explicitly its dependency on Γ. In some applications, it may be interesting to distinguish between positive and negative deviations, so we can also define as

$$\Xi_\Gamma^+ = \{ \quad \xi \quad | \quad ||\xi||_1 \leq \Gamma; \quad 0 \leq \xi \leq 1\}$$

and

$$\Xi_\Gamma^- = \{ \quad \xi \quad | \quad ||\xi||_1 \leq \Gamma; \quad -1 \leq \xi \leq 0\}.$$

Another characteristic of the uncertainty model concerns the probability distribution over the uncertainty set. Although this is an important feature of problems with uncertainty we will not go deeper on this subject, mainly because this work focus on robust optimization, an approach in which we do not need or use a probabilistic distribution over the uncertainty set. For the reader interested in problems involving probabilistic distribution, we suggest [JPTL13], [WLO$^+$11], [LZX$^+$15] and [CH92].

1.3 Solving problems with uncertainty

Let us take a closer look at methods to solve problems with uncertain data. There are, basically, two types of methods dealing with uncertainty in the literature of optimization, namely, stochastic and robust programming. The difference between these two methods lies on the kind of function used to evaluate a feasible solution and the available information on the uncertainty.

Robust programming claims to find a solution that, for a minimization problem, minimizes the cost of the worst evaluation of a feasible solution for all the possible scenarios belonging to the uncertainty set. The stochastic programming method aims to find a solution that has a minimal expected cost over all feasible solutions. More precisely, we can write a *robust minimization problem* as

$$\begin{aligned} \min \quad & \kappa \\ st \quad & \kappa \geq c(x,\xi) \qquad \forall \xi \in \Xi \\ & f_i(x,\xi) \leq b_i(\xi) \quad \forall i \in \{1,\ldots,n\}, \quad \forall \xi \in \Xi \\ & x \in \mathcal{X}. \end{aligned} \tag{1.1}$$

For the stochastic problem, we must have a probabilistic distribution $\mathbb{P}$ over the elements of the uncertainty set to be able to define the expected value of the objective function and the constraints. Mathematically we can express this problem as

$$\begin{aligned} \min \quad & \kappa \\ st \quad & \kappa \geq \mathbb{E}(c(x,\xi)) \qquad \forall \xi \in \Xi \\ & f_i(x,\xi) \leq b_i(\xi) \quad \forall i \in \{1,\ldots,n\}, \quad \forall \xi \in \Xi \\ & x \in \mathcal{X}, \end{aligned} \tag{1.2}$$

where $\mathbb{E}$ denotes the expected value. We recall that such probabilistic distribution is not

needed in the robust problem, which makes robust optimization simpler to apply in cases where limited information is available. Hence, the choice between stochastic optimization and robust optimization depends on the available information on the uncertain parameter and the type of application we are interested in.

1.4 Stochastic programming

As said before, stochastic programming is an approach for modeling optimization problems with uncertainty that assumes the existence of probability distributions governing the uncertainty and that such probabilities are known or can be estimated, by historical data for example. The main feature of such method is to provide solutions that perform well in average. Again, as we said before, the distribution probability of the uncertain parameters plays a central role in all models from stochastic programming.

Stochastic programming can be used when the decisions must be taken only once, or can be adjusted in each period; this kind of problem is usually known as a multi-stage stochastic program. Although stochastic programming can be used in multi-stage problems, the most widely applied and studied stochastic programming problems are two-stage problems. These are problems where the decision maker takes some action in the first stage, after which a random event occurs affecting the outcome of the first-stage decision. Then a second-stage decision can be made in the second stage that compensates for any harmful effects that might have been experienced.

1.5 Robust programming

In this section, we formalize some definitions of robust optimization problems as well as the main methods for solving robust problems or computing heuristic solutions. We pay a particular attention to two approaches: affine decisions rules combined with dualization techniques and decomposition methods. For the reader interested in a more general view

of what has been done in the field we recommend [GMT14] and [GdH15].

Robust optimization seeks to find a solution that is protected against all realizations of the uncertain parameters that lie in an uncertainty set. The motivation for this approach is twofold. First, the model of set-based uncertainty is interesting on its own, and in many applications, it models the uncertainty involved in the application accurately. Second, its computational tractability is often better than the one of stochastic programming approaches [BS07]. The computational tractability of robust optimization explains its success in a broad variety of application areas, especially after the work of Bertismas and Sim [BS04] with the introduction of the budgeted uncertainty set, see also [BS03].

Notice that tractability does not always mean that the robust version conserves the complexity of the nominal problem. For instance, flow problems are difficult even for the budgeted uncertainty set [Min09]. Moreover, even when the uncertainty set is composed of a finite number of scenarios, the robust version can be difficult, for example, the robust shortest path problem is NP-Hard even for an uncertainty set with only two scenarios [GJ98].

In this thesis, we focus on robust linear optimization and robust mixed integer linear optimization, for the reader interested in a more general framework we refer to [BTEGN09].

1.6 Classical approaches in robust optimization

We begin by presenting more formally a robust static problem. We can model a robust static problem as

$$\begin{aligned} \min \quad & c^T x & \\ st \quad & A(\xi)x \leq b(\xi) & \forall \xi \in \Xi \\ & x \in \mathcal{X} & \forall \xi \in \Xi. \end{aligned}$$

These problems represent the most standard class of robust problems. In the following, we present two classical methods to solve this robust static problem: dualization and cutting planes algorithms.

1.6.1 Dualization

We assume that the matrix $A(\xi)$ and the independent vector $b(\xi)$ are affine functions of the uncertain parameters, more precisely, $A_i(\xi) = \bar{A}_i + \hat{A}^i\xi$ and $b(\xi) = \bar{b} + \hat{b}\xi$ where $\bar{A}$, $\hat{A}_i$ and $\hat{b}$ are matrices of appropriate dimensions and $\bar{b}$ is a vector. In this setup we can write the previous problem as

$$\begin{aligned} \min \quad & c^T x \\ st \quad & \bar{A}_i^T x \leq \bar{b}_i + \max_{\xi \in \Xi} (\hat{b}_i^T \xi - (\hat{A}^i \xi)^T x) \qquad \forall i \in 1, \ldots, n \\ & x \in \mathcal{X} \end{aligned}$$

where M_i denotes the i^{th} row of any matrix M.

The idea of the dualization method is to write the inner maximization in each constraint as a minimization problem, thanks to strong duality in linear programming. More precisely, considering a fixed x, we rewrite the maximization problem

$$\max_{\xi \in \Xi} \hat{b}_i^T \xi - (\hat{A}^i \xi)^T x$$

for each $i \in \{1, \ldots, n\}$, where n is the number of rows of the matrix A, recalling that $\Xi = \{ \quad \xi \quad | \quad W\xi \leq q\}$, as

$$\begin{aligned} \min \quad & q^T z \\ st \quad & W^T z \geq \hat{b}_i^T - (\hat{A}^i)^T x. \end{aligned}$$

where z are the dual optimization variables.

Now we can replace the previous maximization problem, of each constraint i by the minimization problem above. Proceeding as said, we obtain the problem given below,

$$\begin{aligned} \min \quad & c^T x \\ st \quad & \bar{A}_i^T x \leq q^T z^i + \bar{b}_i && \forall i \in \{1, \ldots, n\} \\ & W^T z^i \geq \hat{b}_i^T - (\hat{A}^i)^T x && \forall i \in \{1, \ldots, n\} \\ & x \in \mathcal{X} \qquad . \end{aligned}$$

The above problem is a linear program if x is a continuous variable and it can be solved efficiently. Otherwise, it is a mixed integer linear problem.

1.6.2 Cutting Planes

A more general tool for solving robust optimization problems is based on cutting planes methods. In this method, we start with a subset of the original constraints, and we verify in each iteration of the method whether there is a violated constraint. If we find a violated constraint we add the constraint to the model and repeat the process. Otherwise, we stop, and the optimal solution has been found. We assume that the deterministic constraints of the problem are provided in the initial problem.

Formally, we are given an finite set $\Xi^i \subseteq \Xi$ for each constraint i and consider the relaxed master problem RMP defined as

$$\begin{aligned} \min \quad & c^T x \\ st \quad & A_i(\xi) x \leq b_i(\xi) \qquad \forall i \in \{1, \ldots, n\} \quad \forall \xi \in \Xi^i \\ & x \in \mathcal{X}. \end{aligned} \tag{1.3}$$

Given a solution x^*, determining whether there is a violated constraint for x^* comes to

find if there is $i \in \{1, \ldots, n\}$ and $\xi \in \Xi$ such that

$$A_i(\xi)x^* > b_i(\xi).$$

The idea for the algorithm comes straightforward from above. We start first by solving the problem restricted to $\Xi^1 = \Xi^2 = \ldots = \Xi^n = \emptyset$, if the separation problem finds a solution, that is, a pair (i, ξ) such that $A_i(\xi)x^* > b_i(\xi)$, we add ξ to Ξ^i and repeat the process, otherwise we have obtained the optimal solution. The algorithm given below formalizes this idea

Algorithm 1: Cutting plane algorithm for solving classical robust problem.

forall the $j \in \{1, \ldots, n\}$ **do** $\Xi^j \leftarrow \emptyset$;

Solve the problem RMP;

while *Separation problem found pair* (i, ξ) **do**

  $\Xi^i \leftarrow \Xi^i \cup \{\xi\}$;

  Solve the problem RMP;

end

Although the method can be applied to a variety of robust problems, it is effective only when we know how to solve the separation problem efficiently. For instance, if we proceed as before and assume that $A(\xi)$ and $b(\xi)$ are affine functions of the uncertainty parameter, we can solve the separation problem by a simple linear program for each fixed i.

Notice that when $\mathcal{X}$ contains integer restrictions, it is more efficient to use a branch-and-cut aproach instead of the simplistic cutting planes algorithm presented in Algorithm 1

1.7 Robust multi-stage optimization problems

1.7.1 Definition

We address below multi-stage robust problems. We present some definitions and methods used to solve this kind of problem. We begin by recalling a more formal definition of the robust optimization problems. A robust multi-stage problem can be written as:

$$\begin{array}{lll} \min & \kappa & \\ st & \kappa \geq c(\xi)^T x(\xi) & \forall \xi \in \Xi \\ & A(\xi)x(\xi) \leq b(\xi) & \forall \xi \in \Xi \\ & x(\xi) \in \mathcal{X} & \forall \xi \in \Xi. \end{array} \tag{1.4}$$

Where $x(\xi)$ are the optimization variables and $A(\xi)$ and $b(\xi)$ represent the constraints that $x(\xi)$ must satisfy for every ξ belonging to the uncertainty set Ξ and $\mathcal{X}$ represents the deterministic constraints. Below, we consider simplified setting and assume that $A(\xi) = A$ and $c(\xi) = c$ for every $\xi \in \Xi$.

The above formulation does not contain the so-called non-anticipativity constraints recalled next. Let v^i be the projection of the vector v over its first i components, the non-anticipativity constraints impose the following

the *non-anticipativity* constraints

$$\begin{array}{ll} x_i(\xi) = x_i(\xi') & \forall \xi,\xi' \in \Xi, \mathrm{Proj}_{[1\ldots i]}(\xi) = \mathrm{Proj}_{[1\ldots i]}(\xi') \\ y_i(\xi) = y_i(\xi') & \forall \xi,\xi' \in \Xi, \mathrm{Proj}_{[1\ldots i]}(\xi) = \mathrm{Proj}_{[1\ldots i]}(\xi'). \end{array}$$

These constraints represent the fact that the decision maker must take the decisions without any knowledge of the future.

An interesting type of multi-stage problem are two-stage problems. In this kind of problem we consider that the decision must be taken in two distinct periods, x represents

the decisions that must be taken before the uncertain parameters ξ be revealed and $y(\xi)$ represents the decisions that must be made after. We can mathematically model this issue as follows

$$\begin{aligned} \min \quad & \kappa \\ st \quad & \kappa \geq c^T x + g^T y(\xi) \qquad \forall \xi \in \Xi \\ & Ax + Gy(\xi) \leq b(\xi) \quad \forall \xi \in \Xi \\ & x \in \mathcal{X}; y \in \mathcal{Y}. \end{aligned}$$

where $\mathcal{X}$ represents the set of constraints related to the first stage variables, $\mathcal{Y}$ represents the set of constraints related to the second stage variables, G is a matrix and g a vector of appropriated dimensions.

In this context, we can see that the non-anticipativity constraints are not necessary because the variables x are already the same for all the uncertain parameters and the variables y depend on the whole vector $\xi \in \Xi$. The variables x are the so-called here-and-now decisions while the variables y are the wait-and-see decisions.

1.7.2 Solving robust multi-stage problems

Now we present some exact and heuristic methods to solve a multi-stage robust problem. Unlike robust static problems, for multi-stage problems there are no consolidated methods that apply to general problems.

We start by an approximation that is widely used because it preserves, in most cases, the complexity of the problem without uncertainty. Once more, we recall that this is only a short introduction to the field, a more complete review can be found in [BS07].

Affine decision rules

In this relaxation, we assume that the adjustable decisions are not arbitrary functions of the uncertain parameters but rather affine functions of the latter. Of course, this method

can be only applied to continuous variables, hence in this section, we assume that the adjustable variables are continuous variables.

More precisely, $x(\xi) = x_0 + X\xi$, where x_0 is a vector of optimization variables and X is a matrix of optimization variables. Applying this change of variables, we can rewrite the general robust problem as

$$\begin{aligned} \min \quad & \kappa \\ st \quad & \kappa \geq c^T(x_0 + X\xi) && \forall \xi \in \Xi \\ & A(x_0 + X\xi) \leq b(\xi) && \forall \xi \in \Xi \\ & x_0 + X\xi \in \mathcal{X} && \forall \xi \in \Xi. \end{aligned}$$

Notice that, by using this approach, we can turn a multi-stage robust problem into a robust static problem. Although this method considerably reduces the complexity of a multi-stage robust problem one can argue that the approximation obtained is weak for some real applications especially when the adjustable decisions can vary in broad ranges [BG12], [ISS13].

Next, we present methods for solving two-stage robust problems. These methods are based on the dualization technique and the cutting planes method used to static robust optimization problems.

Decompositions methods

The other method that we present is based on the idea of cutting planes for robust problems. Here we allow the fist stage variables to be integer or continuous, although we assume that the second stage variables must remain continuous. We define a finite set of scenarios $\Xi_0 \subseteq \Xi$ and the two-stage relaxed master problem (RMP) associated with it,

given by

$$\begin{aligned} \min \quad & c^T x \\ st \quad & Ax + Gy(\xi) \leq b(\xi) \qquad \forall \xi \in \Xi_0 \\ & x \in \mathcal{X}; y \in \mathcal{Y} \qquad \forall \xi \in \Xi_0, \end{aligned}$$

where the second-stage costs are represented implicitely in the constraints $Ax + Gy(\xi) \leq b(\xi)$. When solving this problem we obtain a first stage solution x^*. To check the feasibility of the first stage solution we need to determine if some constraint is violated in the scenarios belonging to $\Xi \setminus \Xi_0$. This comes to determine if there is a vector $y(\xi^*)$ for some $\xi^* \in \Xi \setminus \Xi_0$ such that

$$Ax + Gy(\xi^*) - b(\xi^*) > 0 \text{ with } y(\xi^*) \in \mathcal{Y}.$$

In [AP16], the authors show that answering to the above question is equivalent to find if the optimal solution of the following problem is smaller than zero or not.

$$\begin{aligned} & \max(A^i x^* + G^i y - b(\xi))\pi \\ st \quad & \xi \in \Xi \\ & y \in \mathcal{Y} \\ & \mathbf{1}\pi = 1 \\ & W^T \pi = 0 \\ & \pi \leq 0, \end{aligned}$$

where π is the vector of dual variables and $\mathbf{1}$ represents the vector with all components equals to 1. The problem presented above can be obtained by using the Farkas's Lemma.

We solve the relaxed master problem restricted to Ξ_0. Then, if the adversarial problem is not zero we add the variables and constraints related to the solution of the adversarial

problem and repeat the process until we have a first-stage solution, that is feasible for every scenario and so optimal. The algorithm below formalizes this idea.

Algorithm 2: Decomposition algorithm for solving the two-stage robust problem.

$\Xi_0 \leftarrow$ arbitrary finite subset Ξ;

$i \leftarrow 0$;

repeat

  Solve RMP. Let $x^\star$ be the obtained solution.;

  Let $\xi^\star$ be the solution of the adversarial problem.;

  $i \leftarrow i + 1$;

  $\Xi_i \leftarrow \Xi_{i-1} \cup \{\xi^\star\}$;

until *the adversarial problem associated with* $x^\star$ *does not admit solution in* $\Xi \setminus \Xi_0$;

Again, the algorithm can be implemented in a branch-and-cut-and-price if $\mathcal{X}$ contains integer restrictions.

Chapter 2

Lot-sizing problems with uncertainty

2.1 Lot-sizing problems

This thesis considers various aspects of robust lot-sizing problems, a well-studied example of multi-stage optimization problem under uncertainty. In the basic version of the lot-sizing problem, one aims to fulfill *the client demands* d_i in each period i inside a planning horizon $H = 1, \ldots, n$ by producing at the period or by some amount produced previously and stored at the lowest possible cost [PW06]. Mathematically we can write the basic lot-sizing problem as

$$
\begin{aligned}
(\mathcal{LS} - \mathcal{U}) \min \quad & \sum_{i=1}^{n} (c_i x_i + h_i s_i) && && (2.1)\\
st \quad & s_i \geq \sum_{j=1}^{i} x_j - d_j && \forall i \in H && (2.2)\\
& \sum_{j=1}^{i} x_j - d_j \geq 0 && \forall i \in H && (2.3)\\
& s_i, x_i \geq 0 && \forall i \in H,
\end{aligned}
$$

where c_i is the unitary production cost at period i and h_i is the unitary storage cost from period i to period $i + 1$. We suppose that we do not have initial stock. The variable x_i

represents the amount produced at period i and s_i represents the amount stored at period i. The objective function (2.1) represents production and storage costs, as expected, and constraint (2.2) represents the amount stored while constraint (2.3) represents the fact that we must satisfy the client demands in each period.

Many features can be added to the basic lot-sizing problem in order to model different practical issues. In this study we consider 3 features:

Backlogging: we are allowed to postpone the client demand to a later period, upon payment of a cost, called *backlogging cost*. To model this feature we can add a variable r_i representing the backlogging at period i with its associated cost p_i in the objective function and the constraints

$$r_i \geq \sum_{j=1}^{i}(d_j - x_j) \text{ and } r_i \geq 0 \quad \forall i \in H, \tag{2.4}$$

Notice that, in order to allow backlog costs one must remove constraints (2.3).

Setup Costs: we must pay a fixed cost g_i for every period $i \in H$ where production takes place. To model this feature we can add binary variables y_i in order to indicate whether or not production takes place and the constraints

$$x_i \leq M y_i \quad \forall i \in H \tag{2.5}$$

where $M \geq \sum_{i=1}^{n} d_i$ is a big constant value.

Production Capacity: we may impose limits C_i to the amount that can be produced in each period $i \in H$. To model this feature it suffices to add the constraints

$$x_i \leq C_i \quad \forall i \in H \tag{2.6}$$

Notice that these features can be mixed. In this work we follow the notation presented in [PW06]. More precisely, we add the subscript *setup* to indicate setup costs, we add

Problem	Setup	Backlog	Capacities	Variables	Constraints
$\mathcal{LS}-\mathcal{U}$	no	no	no	x_i, s_i	(2.2)
$\mathcal{LS}-\mathcal{C}$	no	no	yes	x_i, s_i	(2.2), (2.6)
$\mathcal{BLS}-\mathcal{U}$	no	yes	no	x_i, s_i, r_i	(2.2),(2.4)
$\mathcal{BLS}-\mathcal{C}$	no	yes	yes	x_i, s_i, r_i	(2.2),(2.4), (2.6)
$\mathcal{LS}-\mathcal{U}_{setup}$	yes	no	no	x_i, s_i, y_i	(2.2),(2.5)
$\mathcal{LS}-\mathcal{C}_{setup}$	yes	no	yes	x_i, s_i, y_i	(2.2),(2.5), (2.6)
$\mathcal{BLS}-\mathcal{U}_{setup}$	yes	yes	no	x_i, s_i, r_i, y_i	(2.2),(2.4),(2.5)
$\mathcal{BLS}-\mathcal{C}_{setup}$	yes	yes	yes	x_i, s_i, r_i, y_i	(2.2),(2.4),(2.5), (2.6)

Table 2.1: Different types of lot-sizing problems.

a $\mathcal{B}$ at the beginning of the problem name to indicate that backlog is allowed, and we change $\mathcal{U}$ by $\mathcal{C}$ to indicate the presence of production capacity. In Table 2.1 we present all possible notations for the lot-sizing problems studied.

2.2 Lot-sizing with uncertainty

The primary interest to study lot-sizing problems with uncertainty is that these are the simplest examples of multi-stage problems.

The most natural source of uncertainty is the clients demands d_i in each period because these demands depend on many factors such as client's subjective criteria and current economic situation, which can be very hard to model. Other possible sources of uncertainty are the costs associated with the production, storage, and backlog in each period. In this work, we deal with uncertainty in the clients' demands.

We assume that the clients demands $d_i(\xi)$ in each period i are function of the uncertain parameters ξ. Although some results presented in this thesis can be easily generalized to a general *client demand function* $d_i(\xi)$ at each period, we assume that the client demand function is an affine function $d_i(\xi) = \bar{d}_i + \hat{D}_i\xi$ of the uncertain parameters, where $\hat{d}_i$ is the independent term and $\hat{D}$ is a matrix that represents the correlation between the demands of different periods.

Roughly, we can interpret the term $\bar{d}_i$ as the expected value of the client demands in the period i and each row of the matrix $\hat{D}$ represents the manner in which the vector ξ

influences such expected value at each period. It is common to suppose that the matrix $\hat{D}$ is upper triangular because the demands of future periods do not influence the demands of past periods.

2.3 Stochastic lot-sizing problem

Next, we make a very brief summary of the most known results and experiments involving the stochastic lot-sizing problem. For a more comprehensive and extensive review in this kind of problems we invite the reader to visit chapter 10 in [Tem13] and for more information on stochastic programming we indicate [BL97] as the primary reference.

In the stochastic lot-sizing problem, one usually considers that the client demands are uncertain and that we have a distribution probability for the uncertain demands. One aims to find a solution that fulfills all the demands in every possible scenario and has the minimum average cost.

Recently efficient algorithms have been proposed for several individual cases of the problem, as in [TK04], [HLW01] and [KSLS13]. For example, the stochastic uncapacitated lot-sizing problem with and without setup was studied by Guan and Miller [GM08] and uncapacitated lot-sizing problems without setup costs was studied by Huang and Ahmed [HA09].

Many works have been done on how to formulate stochastic inventory control problems in a dynamic programming framework. These approaches turned out to be effective in characterizing the structure of optimal solution, which can be called *policies*. These solutions have a special structure that allows one to formalize the solution by a simple rule, policy, for example, always produce the demand of client i in period i. For many of these models, it can be shown that *state-dependent* (s,S) policies are optimal [TK04]. The production decision in each period is driven by two thresholds. Specifically, production takes place if and only if the inventory level falls below the threshold s. Also, if production takes places we must produce enough to brought the inventory level up to the threshold

S.

2.4 Robust lot-sizing

In the robust problem, one wants to determine, for each possible realization of the uncertain parameters, the quantities that must be produced, stored, backlogged and if we must have or not production in each period (setups). In this context the variables x,r,s and y become functions of the uncertain parameters, more precisely they become $x(\xi), r(\xi), s(\xi)$ and $y(\xi)$ and the constraints (2.2), (2.4), (2.5) and (2.6) are written for each element of Ξ. Similarly, the objective function aims to minimize the production cost for this solution in a given scenario. More precisely, we present the model for the simple lot-sizing without backlogs and setups.

$$
\begin{aligned}
(\mathcal{LS} - \mathcal{U}(\Xi))\min \quad & \kappa \\
st \quad & \kappa \geq \sum_{i=1}^{n} c_i x_i(\xi) + h_i s_i(\xi) && \forall \xi \in \Xi && (2.7)\\
& s_i(\xi) \geq \sum_{j=1}^{i} x_j(\xi) - d_j(\xi) && \forall i \in H, \xi \in \Xi && (2.8)\\
& \sum_{j=1}^{i} x_j(\xi) - d_j(\xi) \geq 0 && \forall i \in H, \xi \in \Xi && (2.9)\\
& s_i(\xi), x_i(\xi) \geq 0 && \forall i \in H, \xi \in \Xi
\end{aligned}
$$

We should add non-anticipativity constraints to ensure that for two different elements $\dot{\xi}$ and $\ddot{\xi}$ belonging to the uncertainty Ξ that has the same projection at period i, $\dot{\xi}^i = \ddot{\xi}^i$ we must have the same decision taken. Mathematically we can translate this constraint

as the *non-anticipativity* constraints

$$x_i(\xi) = x_i(\xi') \qquad \forall \xi,\xi' \in \Xi, \text{Proj}_{[1\ldots i]}(\xi) = \text{Proj}_{[1\ldots i]}(\xi') \tag{2.10}$$

$$y_i(\xi) = y_i(\xi') \qquad \forall \xi,\xi' \in \Xi, \text{Proj}_{[1\ldots i]}(\xi) = \text{Proj}_{[1\ldots i]}(\xi'). \tag{2.11}$$

where $\text{Proj}_{[1\ldots i]}(\xi)$ denotes the projection of ξ on its first i components.

The other constraints concerning the setups costs, backlog and capacities are presented below. For the backlog, we have that

$$r_i(\xi) \geq \sum_{j=1}^{i}(d_j(\xi) - x_j(\xi)) \text{ and } r_i(\xi) \geq 0 \quad \forall i \in H, \xi \in \Xi. \tag{2.12}$$

For the setup costs we have

$$x_i(\xi) \leq M y_i(\xi) \quad \forall i \in H, \xi \in \Xi. \tag{2.13}$$

For the capacity constraints, we have that

$$x_i(\xi) \leq C_i \quad \forall i \in H, \xi \in \Xi. \tag{2.14}$$

We denote the different robust lot-sizing problems by adding (Ξ) to the notation for the deterministic problem (the problem without uncertainty). To be more precise we adapt Table 2.1 to the robust problem.

Problem	Setup	Backlog	Capacities	Variables	Constraints
$\mathcal{LS} - \mathcal{U}(\Xi)$	no	no	no	x_i, s_i	(2.8), (2.10)
$\mathcal{LS} - \mathcal{C}(\Xi)$	no	no	yes	x_i, s_i	(2.8), (2.14), (2.10)
$\mathcal{BLS} - \mathcal{U}(\Xi)$	no	yes	no	x_i, s_i, r_i	(2.8),(2.12), (2.10)
$\mathcal{BLS} - \mathcal{C}(\Xi)$	no	yes	yes	x_i, s_i, r_i	(2.8),(2.12), (2.14), (2.10)
$\mathcal{LS} - \mathcal{U}_{setup}(\Xi)$	yes	no	no	x_i, s_i, y_i	(2.8),(2.13), (2.10),(2.11)
$\mathcal{LS} - \mathcal{C}_{setup}(\Xi)$	yes	no	yes	x_i, s_i, y_i	(2.8),(2.13), (2.14), (2.10),(2.11)
$\mathcal{BLS} - \mathcal{U}_{setup}(\Xi)$	yes	yes	no	x_i, s_i, r_i, y_i	(2.8),(2.12),(2.13), (2.10),(2.11)
$\mathcal{BLS} - \mathcal{C}_{setup}(\Xi)$	yes	yes	yes	x_i, s_i, r_i, y_i	(2.8),(2.12),(2.13), (2.14), (2.10),(2.11)

Table 2.2: Different types of robust lot-sizing problems.

These problems have been treated in the literature, for example in [Sha11], they are known to be intractable. To the best of our knowledge, no tractable method is known to solve all these robust problems in a satisfactory way.

For this reason, we address particular forms of lot-sizing problems in this thesis. In Chapter 3, we get rid of the non-anticipativity constraints. We end up with a two-stage robust lot-sizing problem. For the two-stage robust lot-sizing problem, the stock and backlog are always wait-and-see variables, while the production and the setup can be here-and-now or wait-and-see variables, depending on the model considered. Notice that if the production is a here-and-now decision, the setup, if it exists, is a here-and-now decision too, because the production is already decided. On the other hand, it is possible that we have a here-and-now setup but a wait-and-see production. Then, we may decide at each period how much we are willing to produce, but the decision if we produce some that day or not must be taken in advance, due to technical restrictions for instance.

In Chapters 4 we simplify the problem by considering models where the production variables are here-and-now decisions. Hence, non-anticipativity constraints only affect the storage and backlog variables and can, therefore, be removed without affecting the problem solution. The hypothesis of these chapters is realistic in an environment where there is no time to adapt the production decisions to the observed demands.

Part II

Main Contributions

This part contains the two main contributions of the thesis. In the first one, we study an appropriated Bender's decomposition method for a robust two-stage problem that has a structure similar to robust lot-sizing problems. We present a dynamic programming algorithm to the adversarial problem as well as an approximation algorithm. The second contribution (Chapter 4) presents a relaxation for the robust lot-sizing problem. More precisely, we relax the non-anticipativity constraints and show the effectiveness of such relaxation. We also study the complexity of the obtained problems.

Chapter 3

A dynamic programming approach for a class of two-stage robust problem

3.1 Introduction

In this chapter we propose a dynamic programming algorithm (DPA) and a unified decomposition algorithm to solve the following optimization problem. Let d be a vector of parameters, z be a vector of optimization variables, μ_i and ν_i be affine functions for each $i = 1, \ldots, n$, and consider the robust constraint

$$f(\psi^{\Sigma}, z) \leq d^T z, \quad \forall \psi \in \Xi, \tag{3.1}$$

with

$$f(\psi^{\Sigma}, z) = \sum_{i=1}^{n} \max\{\mu_i(\psi_i^{\Sigma}, z), \nu_i(\psi_i^{\Sigma}, z)\}. \tag{3.2}$$

and where Ξ gives the uncertainty set and $\psi_i^{\Sigma} = \sum_{j=1}^{i} \psi_j$ denotes the sum of the first i components of uncertain vector ψ for $i = 1, \ldots, n$. Consider a cost vector c and a feasibility set $\mathcal{Z}$. We are interested in solving exactly the following type of robust optimization

problems:

$$\begin{aligned} \min \quad & c^T z \\ \text{s.t.} \quad & z \in \mathcal{Z}, \qquad (3.3) \\ & f(\psi^{\Sigma},z) \leq d^T z, \quad \forall \psi \in \Xi, \qquad (3.4) \end{aligned}$$

where (3.3) represents all constraints not affected by the uncertainty, including the possible integrality restrictions.

Definition (3.2) has its roots in lot sizing problems where, for each period in a given time horizon, one has to pay either storage or backlog cost. It is common to assume demand uncertainty in these problems so that the total demand that must be handled at period i is equal to $\sum_{j=1}^{i} \psi_j$. For the sake of clarity, we delay to a later section of the chapter an important extension of our approach where the elements of the summations $\sum_{j=1}^{i} \psi_j$ involved in (3.2) depend on the optimization variables z.

Our algorithm further requires that the uncertainty set be the budgeted uncertainty polytope Ξ_Γ; for the sake of simplicity we change a little bit the definition of the budgeted uncertainty set to better fit into the algorithm proposed. In this chapter we assume that the budgeted uncertainty set is described by

$$\Xi_\Gamma \equiv \left\{ \psi : \psi_i = \overline{\psi}_i + \widehat{\psi}_i \xi_i, i = 1,\ldots,n, \|\xi\|_\infty \leq 1, \|\xi\|_1 \leq \Gamma \right\}, \qquad (3.5)$$

for positive integers $\widehat{\psi}_i$, $i = 1,\ldots,n$, arbitrary reals $\overline{\psi}_i$, $i = 1,\ldots,n$, and $\Gamma > 0$. The set is extremely popular in robust programming and network optimization and has been used in a wide range of applications. The main purpose of the chapter is to study how to efficiently solve the separation problem for constraint (3.4) and uncertainty set Ξ_Γ, namely

$$(\mathbf{AP}) \qquad \max_{\psi \in \Xi_\Gamma} f(\psi^{\Sigma},z).$$

Traditionally, (**AP**) is either solved through a DPA based on the value of Γ and on

the stock levels [BÖ08], a MILP with big-M coefficients [BÖ08], or approximated via decision rules (e.g. [BGNV05]). More recently, [GdH13] proposed to solve an adversarial separation problem different from (**AP**), which is related to the expansion of the maxima in the definition of f. Differently from these works, we propose here to address (**AP**) via a DPA based on Γ and on $\widehat{\psi}$. Hence, our approach can be applied to a wider range of problems than the seminal work of [BÖ08] that focused on the lot-sizing problem. The worst-case running time of our approach depends on the value of $\widehat{\psi}$ and Γ, yielding a pseudo-polynomial time algorithm. When the deviations are small, our numerical experiments show that the DPA can be orders of magnitude faster than the classical MILP reformulation. Moreover, we show that our DPA gives rise to a FPTAS for (**AP**) and the original robust problem whenever $\mathcal{Z}$ is a convex set and an additional technical assumption is satisfied. We also extend our DPA to combinatorial problems with lower time windows and inventory distribution problems. Notice also that, unlike [GdH13] that considers bi-affine functions μ_i and ν_i, we consider these functions affine herein.

We mention that dynamic programming has already been successfully applied to other RO problems with uncertainty set Ξ_Γ. One of the first works in that aspect is [KN08] which presents a DPA to solve the *robust knapsack problem*. The approach is compared numerically to other solution algorithms in [MPS13] where the authors also study the space complexity of the algorithm. The seminal idea of [KN08] is extended by [Pos14] to any robust combinatorial optimization problems with cost uncertainty whose deterministic counterpart can be solved by a DPA. Differently from [KN08, MPS13, Pos14] which solve the full RO problem by a DPA, the authors of [ACF$^+$13] use a DPA to solve the adversarial problem that arises in robust vehicle routing problems with time windows. A common characteristic of the aforementioned works is that the deterministic version of the problem studied therein (or the AP in case of [ACF$^+$13]) can be solved by a DPA. Thus, these works show how to extend the deterministic DPA to handle the robust versions of the problems.

The algorithm presented herein is different from the above papers on mainly two

aspects. First, like [BÖ08], it does not extend an existing deterministic algorithm because solving (**AP**) is trivial in the deterministic context. Second, the number of states of our algorithm depends on Γ and $\widehat{\psi}$, unlike previous works which only involve Γ. Hence, when each component of $\widehat{\psi}$ is bounded by a polynomial function of the input data, our DPA runs in polynomial time. Otherwise, its running-time is pseudo-polynomial.

3.2 The framework

We describe in this section the general decomposition algorithm used in this chapter. Then, we discuss the complexity of the adversarial problem.

3.2.1 Decomposition

We are interested in solving exactly the following type of robust optimization problems:

$$\begin{aligned} \mathcal{P} \qquad \min \quad & c^T z \\ \text{s.t.} \quad & z \in \mathcal{Z}, && (3.6) \\ & f(\psi^{\Sigma}, z) \leq d^T z, \quad \forall \psi \in \Xi_{\Gamma}. && (3.7) \end{aligned}$$

Constraint (3.6) contains all restrictions not affected by uncertainty, including the possible integrality restrictions on z. Constraint (3.7) is a robust constraint characterized by a function that satisfies (3.2). For the sake of simplicity, we consider a unique robust constraint in $\mathcal{P}$, one readily extends the approach described below to problems with K constraints of type (3.7).

The problem $\mathcal{P}$ contains an infinite number of variables and constraints, making it intractable as such. Here, we tackle the problem by generating a finite subset of variables and constraints on the fly in the course of the decomposition algorithm presented below.

Let $\Xi^0 \subset \Xi_\Gamma$ be a finite set. Since Ξ^0 is finite, we can reformulate

$$f(\psi^\Sigma,z) \leq d^T z, \quad \forall \psi \in \Xi^0, \tag{3.8}$$

as the following finite set of linear inequalities, written for each $\psi \in \Xi^0$:

$$\sum_{i=1}^{n} \varphi_i^\psi \leq d^T z, \tag{3.9}$$

$$\varphi_i^\psi \geq \mu_i(\psi_i^\Sigma,z), \quad \forall i = 1,\ldots,n, \tag{3.10}$$

$$\varphi_i^\psi \geq \nu_i(\psi_i^\Sigma,z), \quad \forall i = 1,\ldots,n, \tag{3.11}$$

where φ^ψ is an additional vector of optimization variables. Our approach is based on the above linear reformulation of (3.8). Specifically, we relax constraints (3.7) for all elements in Ξ_Γ but Ξ^0 and replace robust constraint (3.7) by its linear reformulation, obtaining the Master Problem

$$\begin{aligned}
& \min \quad c^T z \\
(\mathbf{MP}) \qquad & \text{s.t.} \quad z \in \mathcal{Z}, \\
& \qquad \sum_{i=1}^{n} \varphi_i^\psi \leq d^T z, \quad \forall \psi \in \Xi^0, \\
& \qquad \varphi_i^\psi \geq \mu_i(\psi_i^\Sigma,z), \quad \forall \psi \in \Xi^0, i = 1,\ldots,n, \\
& \qquad \varphi_i^\psi \geq \nu_i(\psi_i^\Sigma,z), \quad \forall \psi \in \Xi^0, i = 1,\ldots,n.
\end{aligned}$$

Given a feasible solution z^* to (**MP**), one checks the feasibility of z^* for $\mathcal{P}$ by solving an adversarial problem. Let ψ^* be the optimal solution for the adversarial problem. If $f(\psi^{*\Sigma},z^*) > d^T z^*$, then $\Xi^0 \leftarrow \Xi^0 \cup \{\psi^*\}$, and the corresponding optimization vector φ^{ψ^*} and constraints (3.9)–(3.11) are added to (**MP**). Therefore, the overall algorithm is a row-and-column generation algorithm in the line of those proposed in [ACF$^+$13, ZZ11].

3.2.2 Illustration: Robust Lot-Sizing Problem

Consider the Robust Lot-Sizing Problem (RLSP) defined for a finite planning horizon $\{1,\ldots,n\}$. For each time period $i = 1,\ldots,n$, we are given a capacity C_i, holding cost p_i, a shortage cost s_i, and a production cost c_i. We assume that the vector of demands ψ belongs to set Ξ^{Γ}, where $\overline{\psi}_i$ represents the nominal demand in period i, and $\widehat{\psi}_i$ represents the maximum allowed demand deviation in period i. The amount that needs to be produced in each time period must be decided before the actual demand value is revealed. In contrast, stock levels and backlogged demands are adjusted to each individual demand realization. This model corresponds to $\mathcal{BLS} - \mathcal{C}(\Xi)^x$ in the classification proposed in Chapiter 2.

The problem $\mathcal{BLS} - \mathcal{C}(\Xi)^x$ can be modeled as follows. Variable x_i represents the amount produced at period i and variable θ represents the total storage and backlog costs. For each $\psi \in \Xi$, ψ_i^{Σ} represents the total demand up to time period i for demand vector ψ.

$$\begin{aligned}
\min \quad & c^T x + \theta \\
\text{s.t} \quad & 0 \leq x_i \leq C_i, \quad \forall i = 1,\ldots,n, && (3.12)\\
& \theta \geq \sum_{i=1}^{n} \max\left\{ s_i\left(\psi_i^{\Sigma} - \sum_{j=1}^{i} x_j\right), -p_i\left(\psi_i^{\Sigma} - \sum_{j=1}^{i} x_j\right)\right\}, \quad \forall \psi \in \Xi_{\Gamma}. && (3.13)
\end{aligned}$$

We see readily that the problem is a special case of $\mathcal{P}$ for $z = (x,\theta)$. Namely, $\mathcal{Z}$ is the set defined by (3.12), the components of d corresponding to x and θ are equal to 0 and 1, respectively, and functions f, μ and ν are defined by

$$
\begin{aligned}
f(\psi^{\Sigma},x,\theta) =& \sum_{i=1}^{n} \max\left\{ s_i \left(\psi_i^{\Sigma} - \sum_{j=1}^{i} x_j \right), -p_i \left(\psi_i^{\Sigma} - \sum_{j=1}^{i} x_j \right) \right\}, \\
\mu_i(\psi^{\Sigma},x,\theta) =& s_i(\psi_i^{\Sigma} - \sum_{j=1}^{i} x_j) \text{ and} \\
\nu_i(\psi^{\Sigma},x,\theta) =& - p_i(\psi_i^{\Sigma} - \sum_{j=1}^{i} x_j)
\end{aligned}
$$

Observe that the adversarial problem depends only on the quantities produced. Hence the approach holds if more complex models are considered for the decision problem, such as set-up costs, start-up costs, etc.

3.2.3 Complexity of the adversarial problem

Omitting the dependency on z, the adversarial problem $\max_{\psi \in \Xi_\Gamma} f(\psi^{\Sigma})$ considered in this chapter optimizes a function of the form $f(\psi^{\Sigma}) = \sum_{i=1}^{n} f_i(\psi_i^{\Sigma})$ where f_i is a convex function for each $i = 1,\ldots,n$. Hence, f is convex so that its maximum is attained at least at one of the extreme points of polytope Ξ_Γ, which we denote $\text{ext}(\Xi_\Gamma)$:

$$
\text{ext}(\Xi_\Gamma) \equiv \left\{ \psi : \psi_i = \overline{\psi}_i + \widehat{\psi}_i \xi_i, i = 1,\ldots,n, \xi \in \{-1,0,1\}^n, \|\xi\|_1 \leq \Gamma \right\}.
$$

There exist a few cases in which problem (**AP**) is easy. For instance, when Γ is constant (i.e. it is not part of the input), $\text{ext}(\Xi_\Gamma)$ contains a polynomial number of elements so that (**AP**) can be solved in polynomial time by inspection. More interestingly, the problem can also be solved in polynomial time whenever $\Gamma = n$, which corresponds to Ξ_Γ being a box uncertainty set, see [ISS13]. More recently, the authors of [AD16] have shown that a closely related problem where each function f_i involves only component ψ_i of the uncertain vector is also easy.

While the $\mathcal{NP}$-hardness of (**AP**) is still unknown, simple generalizations of the problem are difficult. For instance, optimizing a general convex function over $\text{ext}(\Xi_\Gamma)$ is

$\mathcal{APX}$-hard, as we present below.

Proposition 1. *Let f be a convex function. Optimization problem $\max_{\psi\in\Xi^\Gamma} f(\psi)$ is $\mathcal{APX}$-hard.*

Proof. Consider the independent set problem on an undirected graph $G = (V,E)$ with n nodes where the goal is to decide whether there is an independent set of size at least $k < n$. Then, the Ξ^Γ be the uncertainty set characterized by $\overline{\psi} = 0$, $\widehat{\psi}_i = 1$ for each $i = 1,\ldots,n$ and $\Gamma = k$, and f be the piecewise linear convex function defined by $f(\psi) = \max_z \left\{ \sum_{i=1}^n \psi_i z_i : z_i + z_j \leq 1, \{i,j\} \in E, z \geq 0 \right\}$. For any $\psi \in \Xi^\Gamma$, $f(\psi) \leq k$. Moreover, G has an independent set of size at least k if and only if $\max_{\psi\in\Xi^\Gamma} f(\psi) = k$. Therefore, the problem of maximizing f over Ξ^Γ is $\mathcal{NP}$-hard.

To show that the problem is $\mathcal{APX}$-hard, consider the independent set problem on 3-regular (cubic) graphs, which is known to be $\mathcal{APX}$-hard [AK00]. Hence, there is a constant $\alpha > 0$ such that it is $\mathcal{NP}$-hard to distinguish whether a cubic graph $G = (V,E)$ has an independent set of size at least k or at most $(1-\alpha)k$. If this is the case that G has an independent set of size at least k, then $\max_{\psi\in\Xi^\Gamma} f(\psi) = k$. Otherwise, if the independent set of G has size at most $(1-\alpha)k$, then one can show that $\max_{\psi\in\Xi^\Gamma} f(\psi) \leq (1-\alpha)k + \frac{3\alpha k}{4} = \left(1 - \frac{\alpha}{4}\right) h$. This implies that maximizing f over Ξ^Γ is $\mathcal{APX}$-hard. □

We study next problems more closely related to (**AP**). We show that if we generalize either the uncertainty set Ξ_Γ or the set of admissible functions f_i, the resulting problem is $\mathcal{NP}$-hard. Namely, consider the following generalization of problem (**AP**):

$$(\widetilde{\mathbf{AP}}) \qquad \max_{\psi\in\widetilde{\Xi}} \sum_{i=1}^n \tilde{f}_i(\psi_i^\Sigma).$$

- If $\widetilde{\Xi}$ is a polytope having a compact description and functions $\tilde{f}_i$ are convex functions of the form $\tilde{f}_i(\psi_i^\Sigma) = \max\{\mu_i(\psi_i^\Sigma),\nu_i(\psi_i^\Sigma)\}$ where μ_i and ν_i are affine functions, then $(\widetilde{\mathbf{AP}})$ is $\mathcal{NP}$-hard. This result is stated in Proposition 2.

- If $\widetilde{\Xi}$ is the set $\text{ext}(\Xi_\Gamma)$ and $\tilde{f}_i$ are general non-negative functions such that $\tilde{f}(\psi^\Sigma) =$

$\sum_{i=1}^{n} \tilde{f}_i(\psi_i^{\Sigma})$ is positive, then ($\widetilde{\mathbf{AP}}$) is $\mathcal{NP}$-hard. This result is stated in Proposition 3.

In view of the above results and of the numerical difficulty of solving problem (**AP**), our conjecture is that the problem is $\mathcal{NP}$-hard.

On the other hand, if the deviations are constant or if their size can be bounded by a polynomial function of the input data then (**AP**) can be solved in polynomial time. This is an immediate consequence of the DPA presented in the next section.

Proposition 2. *Let $\widetilde{\Xi}$ be an arbitrary uncertainty polytope. Optimization problem $\max_{\psi\in\widetilde{\Xi}} f(y(\psi))$ is $\mathcal{NP}$-hard.*

Proof. We consider in this proof function $\hat{f}(\psi) = \sum_{i=1}^{n} \left|\sum_{j=1}^{i} \psi_j\right|$, obtained from (2) by choosing $\mu_i(y_i(\psi)) = y_i(\psi)$ and $\nu_i(y_i(\psi)) = -y_i(\psi)$ for each $i = 1,\ldots,n$ where the dependency on z is omitted for the sake of simplicity. We prove first that problem

$$\max_{\psi\in\widetilde{\Xi}} \hat{f}(\psi) \tag{3.14}$$

has the same optimal solution cost as problem

$$\max_{\psi\in\widetilde{\Xi}'} f^*(\psi), \tag{3.15}$$

with $f^*(\psi) = \sum_{i=1}^{n} |\psi_i|$ and $\widetilde{\Xi}'$ a linear transformation of $\widetilde{\Xi}$. To see this, consider invertible linear transformation

$$\alpha(\psi) = (\psi_1,\psi_1+\psi_2,\psi_1+\psi_2+\psi_3,\ldots,\psi_1+\psi_2+\ldots+\psi_n).$$

Then,

$$\max_{\psi\in\widetilde{\Xi}} \hat{f}(\psi) = \max_{\psi\in\widetilde{\Xi}} \sum_{i=1}^{n} \left|\sum_{j=1}^{i} \psi_j\right| = \max_{\psi\in\widetilde{\Xi}} \sum_{i=1}^{n} |\alpha_i(\psi)| = \max_{\psi\in\widetilde{\Xi}} f^*(\alpha(\psi)) = \max_{\psi\in\alpha(\widetilde{\Xi})} f^*(\psi),$$

and the equality holds for $\widetilde{\Xi}' = \alpha(\widetilde{\Xi})$. Notice that $\widetilde{\Xi}'$ can be described by the same

number of inequalities as $\widetilde{\Xi}$ and these can be computed in polynomial time. Therefore, the $\mathcal{NP}$-hardness of (3.14) follows from the $\mathcal{NP}$-hardness of (3.15), which was proven in [Gus02, Lemma 3.2]. □

Proposition 3. *Let $\tilde{f}_i$ be a non-negative function for each $i = 1,\ldots,n$ and $\tilde{f}(y(\psi)) = \sum_{i=1}^{n} \tilde{f}_i(y_i(\psi))$, with $y_i(\psi) = \sum_{t=1}^{i} \psi_t$, be a positive function. Optimization problem $\max_{\psi \in \text{ext}(\Xi^\Gamma)} \tilde{f}(y(\psi))$ is NP-hard.*

Proof. The decision problem associated with the optimization problem takes the following form: given $\overline{\psi}$,$\widehat{\psi}$, and $\Gamma > 0$, non-negative functions $\tilde{f}_i(y_i(\psi))$, $i = 1,\ldots,n$, and $A \in \mathbb{R}$, is there a $\psi \in ext(\Xi^\Gamma)$ such that $\sum_{i=1}^{n} \tilde{f}_i(y_i(\psi)) \geq A$? We show below that the partition problem can be reduced to the above decision problem. Recall that in the partition problem we are given m positive integers $a_i, i \in M = \{1,\ldots,m\}$ and wish to determine whether there exists a partition $(S, M \setminus S)$ of M such that $\sum_{i \in S} a_i = \sum_{i \in M \setminus S} a_i = \sum_{i \in M} a_i/2$.

For the reduction consider $n = m + 2$, $\overline{\psi}_i = 0, i \in N = \{1,\ldots,n\}, \widehat{\psi}_i = a_i, i \in M, \widehat{\psi}_{m+1} = \widehat{\psi}_{m+2} = 0$. Let $A = \sum_{i=1}^{n} a_i$ and define $\tilde{f}_i(y_i(\psi)) = 0, i \in M$, and

$$\tilde{f}_{m+1}(y_{m+1}(\psi)) = \min\left\{\sum_{i=1}^{n} \psi_i, \frac{A}{2}\right\},\ \tilde{f}_{m+2}(y_{m+2}(\psi)) = \min\left\{A - \sum_{i=1}^{n} \psi_i, \frac{A}{2}\right\}.$$

Hence, from the definition of $\tilde{f}_i$ for $i = 1,\ldots,m+2$, it follows directly that $\sum_{i=1}^{n} f_i(y_i(\psi)) \leq A$. Moreover, the equality holds if and only if $\tilde{f}_{m+1}(y_{m+1}(\psi)) = \tilde{f}_{m+2}(y_{m+2}(\psi)) = A/2$ which is equivalent to find a partition of N with $\sum_{i \in S} a_i = \sum_{i \in M \setminus S} a_i = \frac{A}{2}$ where $S = \{j \in \{1,\ldots,n\} : \psi_j = \widehat{\psi}_j\}$. □

3.3 A dynamic programming algorithm

We present in Section 3.3.1 a simple DPA to solve the adversarial problem for a simplification of Ξ_Γ. We show then in Section 3.3.2 how to approximate (**AP**) and $\mathcal{P}$ through a FPTAS when some additional assumptions are satisfied. We show how to extend the

DPA to the general set Ξ_Γ and to larger classes of functions in Section 3.3.3 and in Section 3.3.4, respectively.

3.3.1 Exact algorithm

For the sake of simplicity, we consider in what follows that Γ is integer and that Ξ does not include downward deviations ($\xi \in \{0,1\}^n$). We discuss in Section 3.3.3 how these simplifications can be relaxed. We further assume that z is fixed throughout the section so that, for each $i = 1, \ldots, n$, we can simplify the writing of affine functions $\mu_i(\psi_i^\Sigma, z)$ and $\nu_i(\psi_i^\Sigma, z)$ by removing the explicit dependency on z. We obtain affine functions

$$\mu_i(\psi_i^\Sigma) = \mu_i^0 + \mu_i^1 \psi_i^\Sigma \quad \text{and} \quad \nu_i(\psi_i^\Sigma) = \nu_i^0 + \nu_i^1 \psi_i^\Sigma,$$

where the dependency on z is hidden in the independent terms μ_i^0 and ν_i^0. Then, we define $f_i(\psi_i^\Sigma) = \max\{\mu_i(\psi_i^\Sigma), \nu_i(\psi_i^\Sigma)\}$, for each $i = 1, \ldots, n$ and $f(\psi^\Sigma) = \sum_{i=1}^n f_i(\psi_i^\Sigma)$.

We are interested here in solving the optimization problem $\max_{\psi \in \Xi_\Gamma} f(\psi^\Sigma)$. Notice that because f is convex, its maximum is always reached at an extreme point of Ξ_Γ, yielding discrete optimization problem

$$\max_{\psi \in \text{ext}(\Xi_\Gamma)} f(\psi^\Sigma). \tag{3.16}$$

Problem (3.16) may be difficult to solve since $\text{ext}(\Xi_\Gamma)$ contains an exponential number of elements and function f is non-linear. However, recall that, because of the definition of Ξ_Γ, we do not need to know the entire vector $\psi \in \text{ext}(\Xi_\Gamma)$ to compute $f(\psi^\Sigma)$. In fact, it is enough to know the cumulative uncertainties $\psi_i^\Sigma = \sum_{t=1}^i \psi_t$ for each $i = 1, \ldots, n$, which are equivalent to the cumulative deviations $\sum_{t=1}^i \psi_t - \sum_{t=1}^i \overline{\psi}_t$ for each $i = 1, \ldots, n$ because $\psi \in [\overline{\psi}, \overline{\psi} + \widehat{\psi}]$. With this in mind, we introduce

$$f_i'(\phi_i) = \max\{\mu_i(\bar{\psi}_i^\Sigma) + \mu_i^1 \phi_i, \nu_i(\bar{\psi}_i^\Sigma) + \nu_i^1 \phi_i\},$$

obtained from f_i by treating separately the cumulative mean $\bar{\psi}_i^\Sigma = \sum_{t=1}^i \overline{\psi}_t$ and the

cumulative deviation $\phi_i = \psi_i^{\Sigma} - \bar{\psi}_i^{\Sigma}$. Namely, let $\xi \in \{0,1\}^n$ be a binary vector that satisfies $\|\xi\|_1 \leq \Gamma$ and let $\psi \in \text{ext}(\Xi_\Gamma)$ be the associated vector of uncertain parameters, defined as $\psi_i = \overline{\psi}_i + \widehat{\psi}_i \xi_i$ for each $i = 1, \ldots, n$. As μ and ν are affine functions, one readily checks that $f_i(\psi_i^{\Sigma}) = f_i'(\phi_i)$ if and only if $\phi_i = \sum_{t=1}^{i} \widehat{\psi}_t \xi_t$. Therefore, adversarial problem (3.16) can be rewritten as

$$
\begin{aligned}
& \max \quad \sum_{i=1}^{n} f_i'(\phi_i) \\
(\mathbf{AP}) \qquad & \text{s.t.} \quad \phi_i = \sum_{t=1}^{i} \widehat{\psi}_t \xi_t, \quad \forall i = 1, \ldots, n, \\
& \qquad \sum_{i=1}^{n} \xi_i \leq \Gamma, \\
& \qquad \xi_i \in \{0,1\}, \quad \forall i = 1, \ldots, n.
\end{aligned}
$$

Up to now we have shown that the optimal solution cost of (**AP**) only depends on the cumulative deviations ϕ_i for each $i = 1, \ldots, n$. To obtain a DPA, we still need a way to enumerate only the most promising cumulative deviations. Let $\overline{\phi}$ be the maximum allowed cumulative deviation, that is, $\overline{\phi} = \max_{S \subseteq \{1,\ldots,n\}: |S| = \Gamma} \sum_{i \in S} \widehat{\psi}_i$. We define $\alpha(j,\gamma,\phi)$, for each triple of integers $1 \leq j \leq n, 0 \leq \gamma \leq \Gamma$ and $0 \leq \phi \leq \overline{\phi}$, as the optimal value of the restricted problem for set $\{1, \ldots, j\}$ with at most γ deviations and a cumulative deviation of ϕ:

$$
\begin{aligned}
\alpha(j,\gamma,\phi) \quad = \quad & \max \quad \sum_{i=1}^{j} f_i'(\phi_i) \\
& \text{s.t.} \quad \phi_j = \phi, && (3.17) \\
& \qquad \phi_i = \sum_{t=1}^{i} \widehat{\psi}_t \xi_t, \quad \forall i = 1, \ldots, j, && (3.18) \\
& \qquad \sum_{i=1}^{j} \xi_i \leq \gamma, && (3.19) \\
& \qquad \xi_i \in \{0,1\}, \quad \forall i = 1, \ldots, j. && (3.20)
\end{aligned}
$$

Let $\alpha(j,\gamma,\phi) = -\infty$ if the feasible set defined by (3.17)−(3.20) is empty because the value of ϕ cannot be reached by a sum of deviations. Hence, we have that $\alpha(1,0,0) = f_1'(0)$, $\alpha(1,\gamma,\widehat{\psi}_1) = f_1'(\widehat{\psi}_1)$ for each $1 \leq \gamma \leq \Gamma$, and $\alpha(1,\gamma,\phi) = -\infty$ for the remaining cases.

We see immediately that the optimal solution cost of the adversarial problem (**AP**), denoted by opt(**AP**), can be computed as $\text{opt}(\mathbf{AP}) = \max_{\phi=0,\ldots,\overline{\phi}} \alpha(n,\Gamma,\phi)$. Moreover, we see easily by contradiction that $\alpha(n,\gamma,\phi)$ satisfies the functional equation stated below.

Lemma 1. *For $j > 1$, each $\alpha(j,\gamma,\phi)$ can be obtained using the following recursion:*

$$\alpha(j,\gamma,\phi) = f_j'(\phi) + \max\{\alpha(j-1,\gamma,\phi), \alpha(j-1,\gamma-1,\phi-\widehat{\psi}_j)\}, \tag{3.21}$$

for each $j = 2,\ldots,n, \gamma = 0,\ldots,\Gamma$, and $\phi = 0,\ldots,\overline{\phi}$.

The computation of $f_j'(\phi)$ can be done in constant time for each $j = 2,\ldots,n$ and $\phi = 0,\ldots,\overline{\phi}$, yielding a pseudo-polynomial worst-case complexity for our DPA.

Lemma 2. *Problem* (**AP**) *can be solved by a DPA in $\mathcal{O}(n\Gamma\overline{\phi})$ operations.*

Using the equivalence between separation and optimization (e.g. [GLS93]), it follows that $\mathcal{P}$ can be solved in pseudo-polynomial time whenever $\mathcal{Z}$ is an *easy* convex set, which is the case for the RLSP.

Corollary 1. *Consider problem $\mathcal{P}$ and let $\mathcal{Z}$ be a convex set that has a polynomial time separation oracle. Then, problem $\mathcal{P}$ can be solved in pseudo-polynomial time.*

Proof. We present next a simple cutting-plane algorithm for solving $\mathcal{P}$. Let J be a non-negative integer and $\mathbf{f}_i^j(z)$ be an affine function of z for each $1 \leq i \leq n, 1 \leq j \leq J$. We solve $\mathcal{P}$ by a cutting-plane algorithm based on the following relaxation:

$$\begin{aligned} & \min \quad c^T z \\ (\mathbf{R}) \qquad & \text{s.t.} \quad z \in \mathcal{Z}, \\ & \qquad \sum_{i=1}^{n} \mathbf{f}_i^j(z) \leq d^T z, \quad \forall j = 1,\ldots,J, \end{aligned}$$

initialized with $J = 0$. Given a feasible solution z^* to (**R**), we solve the adversarial problem. If $\max_{\psi \in \Xi_\Gamma} f(\psi^\Sigma, z^*) > d^T z^*$, we let ψ^* be an optimal solution of the maximization problem, set $J \leftarrow J + 1$, and add a new constraint to (**R**) where $\mathbf{f}_i^j(z) = \mu_i(\psi^{*\Sigma}, z)$ if $\mu_i(\psi^{*\Sigma}, z^*) = \max\{\mu_i(\psi^{*\Sigma}, z^*), \nu_i(\psi^{*\Sigma}, z^*)\}$ and $\mathbf{f}_i^j(z) = \nu_i(\psi^{*\Sigma}, z)$, otherwise. □

Lemma 2 states that solving the adversarial problem using the proposed dynamic approach can be considered an interesting option when the sum of deviations $\overline{\phi}$ is not too large. The cases when the deviations are large correspond to the situations where the uncertain parameters can assume a wide range of values and therefore the decision maker is very conservative or very little is known about the uncertain parameters. We also see that the algorithm is polynomial when Γ is constant since ϕ can take at most n^Γ different values.

3.3.2 Fully polynomial time approximation scheme

We show next how to modify our DPA to obtain a FPTAS for problems $\mathcal{P}$ that satisfy additional assumptions. Our approach works in two steps. First, we adapt to (**AP**) the FPTAS proposed for the knapsack problem by [IK75]. Their main idea is to reduce the precision on the parameters by dividing them with a well-chosen number, identical for all parameters. Our approach holds whenever functions f_i and μ_i satisfy the the technical assumption stated below. Then, we show that whenever $\mathcal{Z}$ is convex, a FPTAS for (**AP**) can be turned into a FPTAS for $\mathcal{P}$.

Assumption 1. *Consider problem* (**AP**) *described by* $(\mu^0, \mu^1, \nu^0, \nu^1, z)$ *and let* $\boldsymbol{\xi} = \max_{i=1,\ldots,n} \widehat{\psi}_i$. *There exists a function* χ *of* $(\mu^0, \mu^1, \nu^0, \nu^1)$ *such that* $LB = \boldsymbol{\xi}\chi(\mu^0, \mu^1, \nu^0, \nu^1)$ *is a lower bound for* opt(**AP**) *and*

$$\frac{\max_{i=1,\ldots,n} \{\mu_i^0, \mu_i^1, \nu_i^0, \nu_i^1\}}{\chi(\mu^0, \mu^1, \nu^0, \nu^1)} \leq \mathcal{P}(n), \tag{3.22}$$

where $\mathcal{P}(n)$ *is a polynomial in* n.

Let us illustrate Assumption 1 on the RLSP, assuming that all components of h and p are positive. For that problem, we see that a lower bound for the optimal solution of the problem is $LB = \frac{\boldsymbol{\xi}\min\{h_n,p_n\}}{2}$, so that (3.22) becomes $\frac{\max\limits_{i=1,\ldots,n}\{h_i,p_i\}}{\min\{h_n,p_n\}} \leq \mathcal{P}(n)$ for some polynomial $\mathcal{P}(n)$. For instance, requiring that $\mathcal{P}(n)$ be equal to some constant $\lambda > 0$ yields the set of instances for which $\frac{\max\limits_{i=1,\ldots,n}\{h_i,p_i\}}{\min\{h_n,p_n\}} \leq \lambda$. Considering polynomials of higher degrees yields a larger set of admissible instances while increasing the computational complexity of the resulting FPTAS.

Lemma 3. *Consider problem* (**AP**) *such that Assumption 1 holds. There exists a FPTAS for* (**AP**).

Proof. For any $\epsilon > 0$, we let $K = \frac{\epsilon\boldsymbol{\xi}\chi(\mu^0,\mu^1,\nu^0,\nu^1)}{2n\Gamma\max\limits_{i=1,\ldots,n}\{\mu_i^0,\mu_i^1,\nu_i^0,\nu_i^1\}}$ and define the new mean value $\overline{\psi}'_i = \frac{\overline{\psi}_i}{K}$ and deviation $\widehat{\psi}'_i = \lfloor\frac{\widehat{\psi}_i}{K}\rfloor$ for each $i = 1,\ldots,n$, and $\overline{\phi}' = \max_{S\subseteq\{1,\ldots,n\}:|S|=\Gamma}\sum_{i\in S}\widehat{\psi}'_i$. Then, execute the DPA presented in the previous section to (**AP**) using the vector of deviations $\widehat{\psi}'$. Using notation $\boldsymbol{\xi}' = \lfloor\frac{\boldsymbol{\xi}}{K}\rfloor$, we see that the running time of the algorithm is polynomial in $(n,\Gamma,1/\epsilon)$ since

$$\mathcal{O}(n\Gamma\overline{\phi}') = \mathcal{O}(n\Gamma^2\boldsymbol{\xi}') = \mathcal{O}\left(n\Gamma^2\left\lfloor\frac{\boldsymbol{\xi}}{K}\right\rfloor\right) = \mathcal{O}\left(n\Gamma^2\left\lfloor\frac{n\Gamma\mathcal{P}(n)}{\epsilon}\right\rfloor\right).$$

We are left to show that the optimal solution to the problem with $\widehat{\psi}'$ is an $(1-\epsilon)$-approximate solution for the original problem.

Let $\xi',\xi^* \in \{\xi|\xi\in\{0,1\}^n,\|\xi\|_1 \leq \Gamma\}$ be the solution returned by the above algorithm and the optimal solution, respectively, and let profit$(\cdot)$ and profit$'(\cdot)$ denote the profit of any element of $\{0,1\}^n$ using deviations $\widehat{\psi}$ and $\widehat{\psi}'$, respectively. Clearly, profit$(\xi') \leq$ opt(**AP**). Then, recall from the definition that K profit$'(\xi) =$

$$\sum_{i=1}^{n}\max\left\{\mu_i^0K + \mu_i^1\sum_{t=1}^{i}\left(K\overline{\psi}_t + \xi_tK\left\lfloor\frac{\widehat{\psi}_t}{K}\right\rfloor\right), \nu_i^0K + \nu_i^1\sum_{t=1}^{i}\left(K\overline{\psi}_t + \xi_tK\left\lfloor\frac{\widehat{\psi}_t}{K}\right\rfloor\right)\right\},$$

for any $\xi \in \{\xi|\xi\in\{0,1\}^n,\|\xi\|_1 \leq \Gamma\}$ and observe that $\widehat{\psi}_t - K\left\lfloor\frac{\widehat{\psi}_t}{K}\right\rfloor \leq K$. Hence, for any

$\xi \in \{\xi | \xi \in \{0,1\}^n, \|\xi\|_1 \leq \Gamma\}$ we have that

$$|\operatorname{profit}(\xi) - K \operatorname{profit}'(\xi)| \leq n\Gamma K \max_{i=1,\ldots,n} \{\mu_i^0, \mu_i^1, \nu_i^0, \nu_i^1\}. \tag{3.23}$$

Therefore,

$$\begin{aligned}
\operatorname{profit}(\xi') &\geq K \operatorname{profit}'(\xi') - n\Gamma K \max_{i=1,\ldots,n} \{\mu_i^0, \mu_i^1, \nu_i^0, \nu_i^1\} \\
&\geq K \operatorname{profit}'(\xi^*) - n\Gamma K \max_{i=1,\ldots,n} \{\mu_i^0, \mu_i^1, \nu_i^0, \nu_i^1\} \\
&\geq \operatorname{profit}(\xi^*) - 2n\Gamma K \max_{i=1,\ldots,n} \{\mu_i^0, \mu_i^1, \nu_i^0, \nu_i^1\} \\
&= \operatorname{opt}(\mathbf{AP}) - \epsilon LB \geq (1-\epsilon)\operatorname{opt}(\mathbf{AP}),
\end{aligned}$$

proving the result. □

The lemma below shows that the existence of a FPTAS for (**AP**) can be translated into a FPTAS for special cases of problem $\mathcal{P}$.

Lemma 4. *Consider the following special case of problem* $\mathcal{P}$

$$\begin{aligned}
& \min \quad c^T z + \theta \\
(\mathbf{P}') \qquad & s.t. \quad z \in \mathcal{Z} \\
& \qquad\quad f(\psi^\Sigma, z) \leq \theta, \quad \forall \psi \in \Xi_\Gamma.
\end{aligned}$$

Assume that $\mathcal{Z}$ *is a convex set that has a polynomial time separation oracle and that there exists a FPTAS for* $\max_{\psi \in \Xi_\Gamma} f$. *There exists a FPTAS for* $(\mathbf{P}')$.

Proof. We must show that for each $\epsilon > 0$, we can provide in polynomial time an $(1+\epsilon)$-approximate solution to $(\mathbf{P}')$. Our approach relies on the cutting-plane algorithm from Corollary 1 with the difference that each (**AP**) is now solved with the FPTAS to provide an $\frac{1}{1+\epsilon}$-approximate solution. Let (z', θ') be the solution returned by the approximate cutting plane algorithm. We claim that $(z', (1+\epsilon)\theta')$ is the desired approximate solution.

Clearly, $(z', (1+\epsilon)\theta')$ is computed in polynomial time. Then, we must verify that

$$\text{opt}(\mathbf{P}') \leq c^T z' + (1+\epsilon)\theta' \leq (1+\epsilon)\text{opt}(\mathbf{P}').$$

To prove the first inequality, we rewrite $(\mathbf{P}')$ as

$$\min_{z \in \mathcal{Z}} c^T z + F(z),$$

where $F(z) = \max_{\psi \in \Xi_\Gamma} f(\psi^\Sigma, z)$. Since θ' is an $\frac{1}{1+\epsilon}$-approximate solution of the corresponding $(\mathbf{AP})$, we have that $\theta' \geq \frac{1}{1+\epsilon} F(z')$. Hence,

$$c^T z' + (1+\epsilon)\theta' \geq c^T z' + F(z') \geq \text{opt}(\mathbf{P}').$$

We prove the second inequality by contradiction. Assuming the inequality does not hold, we obtain:

$$\text{opt}(\mathbf{P}') < \frac{c^T z'}{1+\epsilon} + \frac{1+\epsilon}{1+\epsilon}\theta' \leq c^T z' + \theta'. \tag{3.24}$$

Moreover, because θ' is an approximate solution of the corresponding maximization problem $(\mathbf{AP})$, we have, because z' is optimal.

$$c^T z' + \theta' \leq c^T z' + F(z') \leq \text{opt}(\mathbf{P}'),$$

which is in contradiction with (3.24).

□

3.3.3 General budgeted uncertainty set

We discuss below how to extend the DPA to handle the general uncertainty set Ξ_Γ.

Downward deviations Downward deviations of ψ can be handled by replacing constraints (3.19) and (3.20) in the definition of $\alpha(j, \gamma, \phi)$, by $\sum_{i=1}^{j} |\xi_i| \leq \gamma$ and $\xi_i \in$

$\{-1, 0, 1\}$, respectively. The recursion formula (3.21) is then adapted to:

$$\alpha(j,\gamma,\phi) = f'_j(\phi) + \max\{\alpha(j-1,\gamma,\phi), \alpha(j-1,\gamma-1,\phi-\widehat{\psi}_j), \alpha(j-1,\gamma-1,\phi+\widehat{\psi}_j)\},$$

for each $j = 2,\ldots,n, \gamma = 0,\ldots,\Gamma$, and $\phi = 0,\ldots,\overline{\phi}$. The FPTAS for (**AP**) still holds in this case.

Fractional Γ If Γ is fractional one can take advantage from the fact that the extreme points of Ξ^Γ can have at most one fractional ξ_i. Let $\Gamma^I = \lfloor \Gamma \rfloor$ and $\Gamma^F = \Gamma - \Gamma^I$. Hence (**AP**) can be solved by applying the DPA $n+1$ times, with Γ replaced by Γ^I. In the first iteration, we suppose that ξ has no fractional component (no other change in data is required). In each of the remaining n iterations, we assume that $\xi_j = \Gamma^F$. Then, we redefine $\overline{\psi}_j$ as $\overline{\psi}_j + \Gamma^F \widehat{\psi}_j$ and $\widehat{\psi}_j$ as 0 for that iteration. This approach works because the DPA does not require that $\overline{\psi}_i$ be integer. The FPTAS for (**AP**) also holds in this case.

3.3.4 Other objective functions

We discuss next whether the DPA and the related FPTAS can be extended to more general functions. One the one hand, the DPA holds when $f_i : \mathbb{R} \to \mathbb{R}$ is any quasi-convex function (quasi-convexity is required for replacing Ξ_Γ by $\text{ext}(\Xi_\Gamma)$). Clearly, the computational complexity of the resulting DPA increases according to the computational cost of evaluating each function f_i. For instance, for the type of functions f_i considered in this chapter, this complexity is $O(1)$. A simple example of non-linear function arises in scheduling problems that minimize the squared deviation, where the cost of finishing a job i having a deadline d_i at time ψ_i^Σ is defined as $f_i(\psi^\Sigma) = (\psi_i^\Sigma - d_i)^2$. On the other hand, the FPTAS extends easily only to functions f_i defined as the maxima of K affine functions. This extension could be used to address lot-sizing problems with piecewise linear convex holding costs for instance [TG15]. The extension to K affine functions (instead of 2 as in (3.2)) carries over immediately to problems $\mathcal{P}$ and (**P**′). However, the

row-and-column generation algorithm described in Section 3.2.1 and its cutting-plane version used in Corollary 1 are not designed to handle the more general quasi-convex functions mentioned for the DPA.

The separation problem is not affected by the dependency of f on z, so that the DPA and FPTAS extend directly to more complex functions of z. However, the row-and-column generation algorithm would then involve solving non-linear (**MP**). For instance, allowing μ_i and ν_i to be bi-affine functions, as seen in the recent literature on robust optimization [GdH13, AD16], would lead to bilinear (**MP**) when considering the more general models described in the next section.

3.4 Variable order

3.4.1 Introducing permutations

We discuss in this section how to extend the algorithmic framework discussed in Section 3.2 to handle problems where the order used to define $\psi_i^\Sigma = \sum_{j=1}^{i} \psi_j$ must depend on the values taken by the optimization variables z. Consider for instance the one-machine scheduling problem that minimizes tardiness. Namely, each job i has a given deadline d_i and there is a cost for finishing the job after the deadline that is proportional to the delay. One readily sees that the finishing times of the jobs in a schedule, denoted by z, can be defined by cumulative sums similar to ψ^Σ but depending on the order used in the schedule z. More precisely, the order of the jobs in z can be described by a permutation of the jobs $\{1,\ldots,n\}$. Let $\tau_z(i)$ denote the order of job i in the schedule z. The finishing time of job i is given by

$$y_i(\psi,z) = \sum_{j=1}^{\tau_z(i)} \psi_{\tau_z^{-1}(i)}.$$

Given a penalty weight w_i for each job i, the total penalty cost of the schedule z is

$$f(\psi,z) = \sum_{i=1}^{n} \max\{w_i(y_i(\psi,z) - d_i),0\}.$$

We detail in the next subsection a similar construction for the vehicle routing problem with deadlines. Notice that for more complex problems it is useful to introduce two distinct functions. For instance, in assignment type project scheduling problems [BDM$^+$99], the processing time of a job depends on the amount of resource that is assigned to the job. Hence a solution z would not only describe the order of the jobs, but also their processing times. We would then let τ_z represent the order in which the jobs are realized, while π_z would be a function from $\{1,\ldots,n\}$ to $\{1,\ldots,m\}$ that would characterize the processing times of the jobs among the m available processing times. Notice that our framework does not handle repetitions of the components of ψ in the partial summations so that π_z must be an injective function.

With these applications in mind, we generalize problem $\mathcal{P}$ as follows. We consider two positive integers n and m, such that $m \geq n$ and $\Xi^\Gamma \subset \mathbb{R}^m$ and consider the robust constraint

$$f(y(\psi,z),z) \leq d^T z, \quad \psi \in \Xi^\Gamma, \tag{3.25}$$

with f defined as follows.

Assumption 2. *For every $z \in \mathcal{Z}$, we can define a permutation τ_z of $\{1,\ldots,n\}$ and an injective function $\pi_z : \{1,\ldots,n\} \to \{1,\ldots,m\}$ with $y_i(\psi,z) = \sum_{j=1}^{\tau_z(i)} \psi_{\pi_z(j)}$ such that*

$$f(y(\psi,z),z) = \sum_{i=1}^{n} \max\{\mu_i(y_i(\psi,z),z),\nu_i(y_i(\psi,z),z)\}. \tag{3.26}$$

When z is fixed, we use the shorter notation $f(y(\psi),z)$.

The following extends the algorithm proposed in Section 3.2.1 by considering a vector of optimization variables y^ψ for each $\psi \in \Xi$, which models the function $y(\psi,z)$ in the new master problem. Namely, let us denote by $\mathcal{Y}$ the of solutions that satisfy the linear restrictions that link optimization vectors z and y^ψ such that for any $z \in \mathcal{Z}$ the projection of $\mathcal{Y}$ on y^ψ is reduced to singleton $\{(\sum_{t=1}^{\tau_z(1)} \psi_{\pi_z(t)}, \ldots, \sum_{t=1}^{\tau_z(n)} \psi_{\pi_z(t)})\}$. We illustrate the

set in the next subsection. The counterpart of $\mathcal{P}$ under Assumption 2 is

$$
\begin{aligned}
& \min && c^T z \\
(\mathbf{Pvar}) \qquad & \text{s.t.} && z \in \mathcal{Z}, \\
& && (z,y^\psi) \in \mathcal{Y}, \quad \forall \psi \in \Xi_\Gamma, \\
& && f(y^\psi,z) \leq d^T z, \quad \forall \psi \in \Xi_\Gamma.
\end{aligned} \tag{3.27}
$$

The difference between $\mathcal{P}$ and (**Pvar**) lies in constraints (3.27), which ensure that the optimization vector y^ψ takes the right value for any $z \in \mathcal{Z}$ and $\psi \in \Xi_\Gamma$. The row-and-column-generation algorithm depicted in Section 3.2.1 extends naturally to cope with the new variables and constraints. More precisely, given a feasible solution z^* to (**MP**), one checks the feasibility of z^* for (**Pvar**) by solving an adversarial problem, which is identical to (**AP**) because z^* is fixed. Let ψ^* be the optimal solution for the adversarial problem. If $f(y^{\psi^*},z^*) > d^T z^*$, then $\Xi^0 \leftarrow \Xi^0 \cup \{\psi^*\}$, and the corresponding optimization vectors y^{ψ^*} and φ^{ψ^*} and constraints $(z,y^{\psi^*}) \in \mathcal{Y}$ and

$$
\begin{aligned}
& \sum_{i=1}^{n} \varphi_i^{\psi^*} \leq d^T z \\
& \varphi_i^{\psi^*} \geq \mu_i(y^{\psi^*},z), \quad \forall i = 1,\ldots,n \\
& \varphi_i^{\psi^*} \geq \nu_i(y^{\psi^*},z), \quad \forall i = 1,\ldots,n
\end{aligned}
$$

are added to (**MP**).

We show next that the problems modeled by (**Pvar**) must involve non-connected feasibility sets $\mathcal{Z}$. Roughly speaking, the next result formalizes the intuitive idea that the permutations are non-trivial only if $\mathcal{Z}$ is non-connected, which happens in mixed-integer linear programs for instance.

Lemma 5. *If $\mathcal{Z}$ is a connected set, then for all z,$z' \in \mathcal{Z}$ it holds that*

$$\sum_{t=1}^{\tau_z(i)} \psi_{\pi_z(t)} = \sum_{t=1}^{\tau_{z'}(i)} \psi_{\pi_{z'}(t)}, \text{ for each } i = 1,\ldots,n. \tag{3.28}$$

Proof. Let $\psi \in \text{ext}(\Xi_\Gamma)$ be fixed and let the projection of $\mathcal{Y}$ on its component y be represented by function $\mathbf{Y}(z) = \mathcal{Y} \cap \{z\}$, for all $z \in \mathcal{Z}$. Function $\mathbf{Y}$ is continuous in z, because its image is characterized by constraints that are affine functions of z. Suppose now that $\mathcal{Z}$ is connected and that there exists z,$z' \in \mathcal{Z}$ such that (3.28) does not hold and let $\delta > 0$ be a positive real. Denote by $\overrightarrow{zz'}$ a path in $\mathcal{Z}$ from z to z' and consider a,$b \in \overrightarrow{zz'}$ with $\|a - b\| \leq \delta$ such that (3.28) does not hold. By assumption and recalling that $\widehat{\psi}$ is integer, there exists an index i such that

$$|\mathbf{Y}_i(a) - \mathbf{Y}_i(b)| = \left| \sum_{t=1}^{\tau_a(i)} \psi_{\pi_a(t)} - \sum_{t=1}^{\tau_b(i)} \psi_{\pi_b(t)} \right| \geq 1,$$

proving the discontinuity of $\mathbf{Y}$ at a. □

The lemma implies that when $\mathcal{Z}$ is connected, constraints (3.27) can be removed from (**Pvar**) getting back to the simpler version $\mathcal{P}$.

3.4.2 Illustration: Robust Traveling Salesman Problem with deadlines

Here we consider a variant of the Traveling Salesman Problem where a deadline is considered for the visit to each client. The resulting problem is called the TSPD. The problem occurs in many practical situations and has been studied previously in the stochastic programming literature, see for instance [CT08].

The Robust TSPD, denoted RTSPD, is defined as follows. We are given a complete digraph $G = (V,A)$ with $V = \{0,1,\ldots,n\}$ where node 0 is the depot, costs c_{ij} for crossing arc $(i,j) \in A$, and a deadline b_i associated with each node but the depot. If the vehicle

arrives after time b_i, a penalty cost, denoted by v_i, is incurred per time unit of the violated time. We assume that the traveling times ψ_{ij} are uncertain and belong to the aforementioned budgeted polytope where $\overline{\psi}_{ij}$ is the regular traveling time and $\widehat{\psi}_{ij}$ is the maximum possible delay.

In our formulation below, binary variable x_{ij} indicates whether arc (i,j) is in the solution or not, and continuous variable y_i^ψ represents the time of visit to vertex i when the vector of traveling times ψ is considered. We assume node 0 is the depot, from where the vehicle departs. The RTSPD can be modeled as follows.

$$\min \quad \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} + \theta \tag{3.29}$$

$$\text{s.t.} \quad \sum_{i=1}^{n} x_{ij} = 1, \quad \forall j \in V, \tag{3.30}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i \in V, \tag{3.31}$$

$$y_0^\psi = 0, \quad \forall \psi \in \Xi_\Gamma, \tag{3.32}$$

$$y_j^\psi \geq y_i^\psi + \psi_{ij} - M(1 - x_{ij}), \quad \forall \psi \in \Xi_\Gamma, (i,j) \in A, \tag{3.33}$$

$$\theta \geq \sum_{i=1}^{n} \max\{v_i(y_i^\psi - b_i), 0\}, \quad \forall \psi \in \Xi_\Gamma, \tag{3.34}$$

$$x_{ij} \in \{0,1\}, \quad \forall (i,j) \in A, \tag{3.35}$$

where M is a big-M constant. Constraints (3.30) and (3.31) ensure that each node is visited once. Constraints (3.32) and (3.33) define the time variables for the visit to each node, which also forbids cycles. Constraints (3.34) models the penalty incurred for not meeting the deadline at each node, and adds the penalty to θ.

We have yet to show that the above formulation is a special case of (**Pvar**). Here, set $\mathcal{Z}$ contains all $z = (x,\theta)$ where θ is an unrestricted real and x is a Hamiltonian cycle, that is, a binary vector that satisfies constraints (3.30) and (3.31) and whose associated subgraph does not contain cycles. Strictly speaking, the above formulation is not a special case of (**Pvar**) because cycles are forbidden through constraints (3.33), which are

constraints that characterize $\mathcal{Y}$ and should therefore not contribute to the definition of $\mathcal{Z}$ as introduced in Section 3.4.1. Hence, to be truly coherent with (**Pvar**), one should add to the model classical constraints to forbid cycles (e.g. subtour inequalities, cut inequalities, MTZ inequalities, ...), which we omit to keep our model as simple as possible. Set $\mathcal{Y}$ is then defined by constraints (3.32) and (3.33). Namely, consider a fixed $z = (x,\theta) \in \mathcal{Z}$ where x is a fixed Hamiltomian cycle, and let $\tau_z(i)$ denote the position of node i in the cycle, starting from the depot ($\tau_z(0) = 0$), and let $\pi_z(i)$ denote the arc that comes in position i in the cycle. One readily verifies that (3.32) and (3.33) yield the following restriction for y_i^ψ for each $i = 1,\ldots,n$:

$$y_i^\psi \geq \sum_{t=1}^{\tau_z(i)} \psi_{\pi_z(i)}. \tag{3.36}$$

Again, to be faithful to (**Pvar**), constraint (3.36) should hold to equality. Although this could be enforced by complementing the formulation with additional linear constraints with big-M constants, these are unnecessary because the equality holds in any optimal solution of the RTSPD. Finally, function $f(y(\psi,z),z)$ can be defined from the rhs of (3.34), $\sum_{i=1}^{n} \max\{v_i(y_i(\psi,z)-b_i), 0\}$, which satisfies Assumption 2. Therefore, the decomposition of the problem can be done similarly to Section 3.2. Given a solution $z \in \mathcal{Z}$ to the master problem, the adversarial problem looks for the scenario ψ that maximizes $f(y(\psi),z)$, namely,

$$\text{opt}(\mathbf{AP}) = \max_{\psi \in \Xi_\Gamma} \left\{ \sum_{i=1}^{n} f_i(\psi) \right\} \text{ where } f_i(\psi) = \max\{v_i(y_i(\psi) - b_i), 0\}.$$

Differently from the RLSP presented in the previous subsection, we see that the meaning of $y_i(\psi,z)$ depends here on vector $z = (x,\theta)$, since the order in which the nodes and arcs are visited depends on the specific Hamiltonian cycle x.

3.5 Extensions

In this section we discuss two extensions of the adversarial problem and the DPA. For the sake of simplicity, these extensions are presented for specific optimization problems.

3.5.1 A simple inventory distribution problem

In this section we consider an example of the case where the adversarial problem can be separated into k subproblems with a single linking constraint which is the constraint imposing a maximum number of Γ deviations. Each subproblem coincides with (**AP**). The approach consists in solving each subproblem (**AP**) with the DPA for each possible number of deviations and then, in a second stage, combine the k subproblems.

We exemplify this situation below with a robust variant of a simplistic inventory distribution problem. We are given a set k of retailers and a set of n time periods. The company controls the inventory at the retailers and needs to decide when and how much to deliver in each time period at each retailer. We define the following parameters: C is the distribution capacity per period for each retailer and $\psi_{ij}, s_{ij}, p_{ij}, f_{ij}$ represent for retailer j in period i, the demand, backlogging cost, holding cost and fixed transportation cost, respectively. As before, demand vector ψ is uncertain and belong to uncertainty set Ξ_{IR}^{Γ} defined as $\Xi_{IR}^{\Gamma} = \left\{\psi : \psi_{ij} = \overline{\psi}_{ij} + \widehat{\psi}_{ij}\xi_{ij}, i = 1,\ldots,n, j = 1,\ldots,k, \|\xi\|_{\infty} \leq 1, \|\xi\|_1 \leq \Gamma\right\}$. Hence, Γ is the total number of deviations allowed.

In our simplistic variant presented below, the only distribution variables are the continuous variables x_{ij} that describe the quantity delivered to retailer j in period i. Hence, x_{ij} plays the role of the former production variable x_i used in Section 3.2.2. A formulation

for the Robust Inventory Routing Problem $RIRP$ follows.

$$\begin{aligned}
\min \quad & f^T x + \theta \\
\text{s.t} \quad & 0 \leq x_{ij} \leq C, \quad \forall i = 1, \ldots, n, j = 1, \ldots, k, \\
& \theta \geq \sum_{i=1}^{n} \sum_{j=1}^{k} \max \left\{ s_{ij} \sum_{t=1}^{i} \left(\psi_{tj} - x_{tj} \right), -p_{ij} \sum_{t=1}^{i} \left(\psi_{tj} - x_{tj} \right) \right\}, \quad \forall \psi \in \Xi_{IR}^{\Gamma}. \quad (3.37)
\end{aligned}$$

Unlike the simple lot-sizing from Section 3.2.2, it is not possible here to sum up total demands up to period i in variable y_i^{ψ} since the latter depends also on the particular retailer. One way to avoid this difficulty is to ignore the correlation of demands for different retailers, replacing the above model with

$$\begin{aligned}
\min \quad & f^T x + \sum_{j=1}^{k} \theta_j \\
\text{s.t} \quad & 0 \leq x_{ij} \leq C, \quad \forall i = 1, \ldots, n, j = 1, \ldots, k, \\
& \theta_j \geq \sum_{i=1}^{n} \max \left\{ s_{ij} \left(\psi_i^{\Sigma} - \sum_{t=1}^{i} x_{tj} \right), -p_{ij} \left(\psi_i^{\Sigma} - \sum_{t=1}^{i} x_{tj} \right) \right\}, \\
& \qquad \forall j = 1, \ldots, k, \psi \in \Xi_{uncor}^{\Gamma^j},
\end{aligned}$$

with $\Xi_{uncor}^{\Gamma^j} = \left\{ \psi : \psi_i = \overline{\psi}_i + \widehat{\psi}_i \xi_i, i = 1, \ldots, n, \|\xi\|_{\infty} \leq 1, \|\xi\|_1 \leq \Gamma^j \right\}$, and $\psi_i^{\Sigma} = \sum_{t=1}^{i} \psi_t$.

The uncorrelated model yields k adversarial problems, each of them identical to the adversarial problem of the RLSP. Still, it may not be satisfactory in some applications to ignore the correlations of demands among different retailers. Hence, we explain below how to solve the adversarial problem induced by constraint (3.37). As mentioned previously, for each retailer $j \in \{1, \ldots, k\}$, we have an adversarial problem which coincides with the adversarial problem of the RLSP. The k problems are linked through the maximum number of deviations Γ. For each $j \in \{1, \ldots, m\}$, $\gamma \in \{0, \ldots, \Gamma\}$ let $\delta_{j\gamma}$ denote the value of the adversarial problem for the RLSP for retailer j with γ deviations, computed by the DPA from Section 3.3. The value of the adversarial problem for the RIRP is given

by the best combination of these k subproblems:

$$\text{opt}(\mathbf{AP}') = \max \left\{ \sum_{j=1}^{k} \sum_{\gamma=0}^{\Gamma} \delta_{j\gamma} u_{j\gamma} : \sum_{j=1}^{k} \sum_{\gamma=0}^{\Gamma} \gamma u_{j\gamma} \leq \Gamma, u_{j\gamma} \in \{0,1\}, \right.$$
$$\left. j = 1, \ldots, k, \gamma = 0, \ldots, \Gamma \right\}.$$

where $u_{j\gamma}$ is a binary variable that indicates whether γ deviations are considered for retailer j. This linking problem, known as the Linking Set Problem can be solved by dynamic programming in $\mathcal{O}(k\Gamma)$ operations, see [AR09]. Recall that $\overline{\phi} = \max_{S \subseteq \{1,\ldots,n\}:|S|=\Gamma} \sum_{i \in S} \widehat{\psi}_i$. We obtain the following result.

Lemma 6. *Problem* $(\mathbf{AP}')$ *can be solved by a DPA in* $\mathcal{O}(n\Gamma\overline{\phi} + k\Gamma)$ *operations.*

Notice that, for the sake of clarity, we presented a simplistic inventory routing problem. However, our approach extends directly to inventory routing problems where the routing and/or distribution is modeled by introducing binary variables (e.g. [SCL12]), as well as problems with different capacities, since these refinements do not affect $(\mathbf{AP}')$. We also point out that similar adversarial problems occur in other problems such as the vehicle routing problems with deadlines with uncertain traveling times where the number of delays is bounded by Γ. In the latter problem, it can be important to bound the total number of delays for all vehicle by a single value of Γ.

3.5.2 The TSP with time-windows

The RTSPD considered in the previous section is a simplification of the widely studied TSP where a full time window $[a_i, b_i]$ is associated to each node different from the depot, thus complementing the deadline b_i with a lower limit a_i. The difficulty of handling a full time window, instead of a deadline, depends on the type of constraints considered: soft or hard, where a soft constraint can be violated at some cost, and a hard constraint must always be satisfied. For instance, the deadline introduced for the RTSPD in the previous

section is a soft constraint. A hard left constraint imposes that if the salesman arrives at node before a_i he must wait until a_i. We examine next the four different combinations for left and right time limits. If left and right time limits are soft, with unitary penalty costs denoted by u and v, respectively, the resulting problem can be formulated as the RTSPD studied previously replacing constraint (3.34) with

$$\theta \geq \sum_{i=1}^{n} \max\{u_i(y_i^\psi - b_i), -v_i(y_i^\psi - a_i)\}, \quad \forall \psi \in \Xi_\Gamma.$$

One readily sees that the above robust constraint is defined by a function that satisfies Assumption 2. If left time limit is soft, with unitary penalty cost p, and right time limit is hard, the problem can be formulated as the RTSPD studied previously replacing constraint (3.34) with robust constraints

$$\theta \geq \sum_{i=1}^{n} \max\{0, -v_i(y_i^\psi - a_i)\}, \quad \forall \psi \in \Xi_\Gamma, \tag{3.38}$$

$$y_i^\psi \leq b_i, \qquad \forall i = 1, \ldots, n, \psi \in \Xi_\Gamma. \tag{3.39}$$

One readily sees that both robust constraints above are defined by functions that satisfy Assumption 2. Actually, solving the (**AP**) associated to constraint (3.39) can be done in polynomial time since it comes down to computing the Γ highest components of $\widehat{\psi}$ among $\{\widehat{\psi}_1, \ldots, \widehat{\psi}_i\}$ for each $i = 1, \ldots, n$. If both time limits are hard, the problem does not enter the framework studied in this chapter. The authors of [ACF$^+$13] propose different approaches for that problem and show that the corresponding (**AP**) can be solved in polynomial time. The last case, denoted RTSPTW in what follows (standing for Traveling Salesman Problem with Time Windows), considers hard left time limit and soft right time limit with unitary penalty cost p (see [ACD13] for an application). This case is more complex than the previous ones. Still, we show in the balance of the section that the associated adversarial problem can also be solved by a DPA in pseudo-polynomial time.

The mathematical model for the RTSPTW is close to the one given for the TSPD. In addition to (3.29) – (3.35), we must consider a new constraint imposing a lower bound on the start time of each visit:

$$y_i^{\psi} \geq a_i \quad \forall \psi \in \Xi_{\Gamma}, i \in V. \tag{3.40}$$

Unfortunately, the addition of constraints (3.40) to the formulation of the RTSPD prevents the RTSPTW from being a special case of problem (**Pvar**). Namely, let $z = (\theta,x) \in \mathcal{Z}$ where x is the incidence vector of a Hamiltonian cycle and θ the worst-case cost due to the delay, and let τ_z and π_z be the functions defined in Section 3.4.2. To avoid carrying the functions throughout, we assume without loss of generality that $\tau_z(i) = i$ and $\pi_z(i) = i$ for each $i = 1,\ldots,n$. Constraints (3.40) break the structure witnessed for the RTSPD since the arrival time $y_i(\psi)$ at node i can no longer be defined as an affine function of ψ, such as (3.36) used for the RTSPD. Namely, the wait of the salesman in case he arrives at node i before a_i yields arrival times that can be formulated recursively as

$$y_i(\psi) = \max(a_i, a_{i-1} + \psi_{(i-1)i}), \tag{3.41}$$

for each $i = 1,\ldots,n$, where we recall that $(i-1,i)$ denotes the arc that enters i and leaves its predecessor $i-1$ in the Hamiltonian cycle. Therefore, penalty function $f(y(\psi),z) = \sum_{i=1}^{n} \max\{u_i(y_i(\psi) - b_i), 0\}$, with $y_i(\psi)$ defined in (3.41), does not satisfy Assumption 2 because y is not affine. Fortunately, it is possible to adapt our DPA to handle this difficulty. For the sake of simplicity, we explain the generalization of the DPA directly for the RTSPTW, rather than using the general framework depicted in Section 3.4.

Expanding recursively the maxima in (3.41), $y_i(\psi)$ can be expressed as (see [ACF+13])

$$y_i(\psi) = \max_{\ell=1,\ldots,i} \left\{ a_\ell + \sum_{t=\ell}^{i-1} \psi_{t(t+1)} \right\}. \tag{3.42}$$

As for the RTSPD, our objective penalizes the violation of the right time window b_i

with unitary penalty cost h_i. Hence the adversarial problem is in this case $\text{opt}(\mathbf{AP}^*) = \max_{\psi\in\Xi_\Gamma}\left\{\sum_{i=1}^{n} f_i(\psi)\right\}$ where

$$f_i(\psi) = v_i\left[\max_{\ell=1,\ldots,i}\left\{a_\ell + \sum_{t=\ell}^{i-1}\psi_{t(t+1)}\right\} - b_i\right]^+ = \max_{\ell=1,\ldots,i}\left\{u_i\left[a_\ell + \sum_{t=\ell}^{i-1}\psi_{t(t+1)} - b_i\right]^+\right\},$$

where we used the simplified notation $[x]^+$ for $\max\{0,x\}$. Let $\psi_{[\ell i]}$ denote the subvector $\{\psi_{t(t+1)} : t = \ell,\ldots,i-1\}$ and define $\overline{f}_{\ell i}(\psi_{[\ell i]}) = v_i[a_\ell - b_i + \sum_{t=\ell}^{i-1}\psi_{t(t+1)}]^+$. Hence, $f_i(\psi) = \max_{\ell=1,\ldots,i}\overline{f}_{\ell i}(\psi_{[\ell i]})$. Let $\beta(m,\gamma)$ be the value of the optimal solution of the restricted problem defined for the subpath $1,\ldots,m$ with at most γ deviations:

$$\beta(m,\gamma) = \max_{\psi\in\Xi^\gamma_{[1m]}}\left\{\sum_{i=1}^{m}\max_{\ell=1,\ldots,i}\overline{f}_{\ell i}(\psi_{[\ell i]})\right\},$$

where $\Xi^\gamma_{[\ell i]} \equiv \left\{\psi : \psi_t = \overline{\psi}_t + \widehat{\psi}_t\xi_t, t = \ell,\ldots,i, \xi\in\{0,1\}^{i-\ell+1}, \|\xi\|_1 \le \gamma\right\}$. Clearly, $\text{opt}(\mathbf{AP}^*) = \beta(n,\Gamma)$.

The rest of the section is devoted to the construction of a DPA to compute $\beta(n,\Gamma)$. Notice that for any t, $\sum_{i=t}^{m}\overline{f}_{ti}$ satisfies (3.2), so that the sum can be optimized over the set $\Xi^\gamma_{[tm]}$ in pseudo-polynomial time by appling the algorithm presented in Section 3.3.1. Let us denote $\overline{f}^\beta(\xi_{[1m]}) = \sum_{i=1}^{m}\max_{\ell=1,\ldots,i}\overline{f}_{\ell i}(\psi_{[\ell i]})$ so that $\beta(m,\gamma)$ can be rewritten as

$$\beta(m,\gamma) = \max_{\psi\in\Xi^\gamma_{[1m]}}\overline{f}^\beta(\xi_{[1m]}).$$

The algorithm from Section 3.3.1 cannot be used directly to optimize $\overline{f}^\beta(\xi_{[1m]})$ because of the maximization involved in the definition of $\overline{f}^\beta(\xi_{[1m]})$. Hence, we used an alternative recursion based on the key lemma below. The lemma expresses $\overline{f}^\beta(\xi_{[1m]})$ from the set of functions $\left\{\overline{f}^\beta(\xi_{[1t]}) : 1\le t\le m-1\right\}$ and the sums $\left\{\sum_{i=t}^{m}\overline{f}_{ti}(\psi_{[ti]}) : 1\le t\le m\right\}$. We show in the balance of the section how this leads to a DPA.

Lemma 7. *Let $\psi\in\Xi^\gamma$ be fixed and $m\in\{2\ldots,n\}$. It holds that*

$$\overline{f}^{\beta}(\xi_{[1m]}) = \max_{t=1,\ldots,m} \left\{ \overline{f}^{\beta}(\xi_{[1(t-1)]}) + \sum_{i=t}^{m} \overline{f}_{ti}(\psi_{[ti]}) \right\}.$$

Proof. Since ψ is fixed, we simplify notations and denote $\overline{f}_{\ell i}(\psi_{[\ell i]})$ as $\overline{f}_{\ell i}$ in the rest of the proof. Notice that from the definition of $\overline{f}_{\ell m}$, the following holds for each $i \in \{\ell, \ldots, m\}$:

$$\arg\max_{\ell=1,\ldots,i} \overline{f}_{\ell m} = \arg\max_{\ell=1,\ldots,i} \overline{f}_{\ell i}.$$

Therefore, if $\arg\max_{\ell=1,\ldots,m} \overline{f}_{\ell m} = t$ and $t \leq i$, then $\arg\max_{\ell=1,\ldots,i} \overline{f}_{\ell i} = t$. This can be equivalently written as

$$\overline{f}_{tm} = \max_{\ell=1,\ldots,m} \overline{f}_{\ell m} \quad \Rightarrow \quad \overline{f}_{ti} = \max_{\ell=1,\ldots,i} \overline{f}_{\ell i} \text{ for all } i = t, \ldots, m. \tag{3.43}$$

The counterpart of (3.43) for the whole sum $\sum_{i=1}^{m} \max_{\ell=1,\ldots,i} \overline{f}_{\ell i}$ is

$$\overline{f}_{tm} = \max_{\ell=1,\ldots,m} \overline{f}_{\ell m} \quad \Rightarrow \quad \sum_{i=1}^{m} \max_{\ell=1,\ldots,i} \overline{f}_{\ell i} = \sum_{i=1}^{t-1} \max_{\ell=1,\ldots,i} \overline{f}_{\ell i} + \sum_{i=t}^{m} \overline{f}_{ti},$$

and the result follows by taking the maximum over all $t = 1, \ldots, m$ because we do not know in advance which corresponds to $\arg\max_{\ell=1,\ldots,m} \overline{f}_{\ell m}$. □

Using Lemma 7 we have, for each $2 \leq m \leq n$ and $0 \leq \gamma \leq \Gamma$:

$$\begin{aligned}
\beta(m,\gamma) &= \max_{\psi \in \Xi^{\gamma}_{[1m]}} \left\{ \overline{f}^{\beta}(\xi_{[1m]}) \right\} \\
&= \max_{\psi \in \Xi^{\gamma}_{[1m]}} \max_{t=1,\ldots,m} \left\{ \overline{f}^{\beta}(\xi_{[1(t-1)]}) + \sum_{i=t}^{m} \overline{f}_{ti}(\psi_{[ti]}) \right\} \\
&= \max_{t=1,\ldots,m} \max_{\delta=0,\ldots,\gamma} \left\{ \max_{\psi \in \Xi^{\delta}_{[1(t-1)]}} \overline{f}^{\beta}(\xi_{[1(t-1)]}) + \max_{\psi \in \Xi^{\gamma-\delta}_{[tm]}} \sum_{i=t}^{m} \overline{f}_{ti}(\psi_{[ti]}) \right\} \\
&= \max_{t=1,\ldots,m} \max_{\delta=0,\ldots,\gamma} \left\{ \beta(t-1,\delta) + \max_{\psi \in \Xi^{\gamma-\delta}_{[tm]}} \sum_{i=t}^{m} \overline{f}_{ti}(\psi_{[ti]}) \right\} \\
&= \max_{t=1,\ldots,m} \max_{\delta=0,\ldots,\gamma} \left\{ \beta(t-1,\delta) + \overline{F}(t,m,\gamma-\delta) \right\},
\end{aligned} \tag{3.44}$$

where $\overline{F}(t,m,\gamma-\delta) = \max_{\psi\in\Xi^{\gamma-\delta}_{[tm]}} \sum_{i=t}^{m} \overline{f}_{ti}(\psi_{[ti]})$. Furthermore, for $m = 1$ and each $\gamma \in \{0,\ldots,\Gamma\}$, we have

$$\beta(1,\gamma) = \max_{\psi\in\Xi^{\gamma}_{[11]}} \overline{f}_{11}(\psi_{[11]}) = v_1[a_1 - b_1]^+ = 0. \tag{3.45}$$

Combining (3.44) and (3.45), we obtain a DPA to solve (**AP***).

We conclude the section by showing that (3.44) yields a pseudo-polynomial DPA. Notice that function

$$\bar{f}_{tm}(\psi_{[tm]}) = \sum_{i=t}^{m} \overline{f}_{ti}(\psi_{[ti]})$$

satisfies Assumption 2 for each $1 \leq t \leq m \leq n$. Hence, we can apply the DPA from Section 3.3 to compute $\overline{F}(t,m,\gamma) = \max_{\psi\in\Xi^{\gamma}_{[tm]}} \bar{f}_{tm}(\psi_{[tm]})$ for each $1 \leq t \leq m \leq n$ and $0 \leq \gamma \leq \Gamma$. Namely, let α_t be the table used to compute $\overline{F}(t,n,\Gamma)$ through the DPA from Section 3.3. We readily see that

$$\overline{F}(t,m,\gamma) = \max_{\phi=0,\ldots,\overline{\phi}} \alpha_t(m-t,\gamma,\phi),$$

for each $1 \leq t \leq m \leq n$ and $0 \leq \gamma \leq \Gamma$. Therefore, we can obtain all values in $\{\overline{F}(t,m,\gamma) : 1 \leq t \leq m \leq n, 0 \leq \gamma \leq \Gamma\}$ by applying the DPA from Section 3.3 to $\overline{F}(t,n,\Gamma)$ for each $1 \leq t \leq n$. This yields a computational time of $O(n^2\Gamma\overline{\phi})$ which is done in a pre-processing phase. Once all values of $\overline{F}$ have been computed, (3.44) can be solved in $O(n^2\Gamma^2)$. Since $\Gamma \leq \overline{\phi}$, we obtain the following worst-case complexity.

Lemma 8. *Problem* (**AP***) *can be solved by a DPA in* $\mathcal{O}(n^2\Gamma\overline{\phi})$ *operations.*

We finish the section by noticing that constraints similar to time windows appear also in scheduling problem with release times and deadlines [Tsa03]. In the deterministic version of the problem, we consider a set $N = \{1,\ldots,n\}$ of tasks. For each task $i \in N$ we are given a release time a_i, which is the time at which task i can start being processed, a deadline b_i and a duration ψ_i. The objective of the problem is to find a feasible schedule of the tasks in a single machine such that the weighted sum of the deadline violations are minimized. In the robust approach, the durations of the tasks are uncertain and belongs to

n	$\Gamma^{0.10}$	$\Gamma^{0.05}$	$\Gamma^{0.01}$
50	11	13	18
100	14	18	24
200	20	25	34

Table 3.1: Values of Γ^ϵ obtained by rounding up the values prescribed by the probabilistic bound from [BS04].

uncertainty set Ξ_Γ. Similarly to the discussion made for the traveling salesman problem, one can consider different versions of the problem depending on weather the release times and deadlines are soft or hard constraints. Then one readily sees that for each of these variants, the adversarial problem is similar to the aforementioned adversarial problems.

3.6 Computational experiments

We compare in this section our DPA and the classical MIP formulation for solving the lot-sizing problem with row-and-column generation algorithms.

3.6.1 Instances and details

We consider three numbers of periods: 50, 100, and 200. For each one of them we create four sets of instances: $S1$, $S2$, $S3$ and $S4$. For all sets we generate the storage cost for each period randomly and uniformly from an interval [5,10]. The difference between the sets lies in the backlog cost. For each $i \in \{1,2,3,4\}$, instances in Si are defined by backlog cost equal to i times the storage cost in each period. For all instances the nominal demand is generated randomly from interval $[50, 100]$. Then, we consider five levels of deviations, ranging from 10% to 50% of the nominal demand. We round up the obtained deviations to ensure that they are integer. Finally, we also consider three different values for the budget of uncertainty Γ, motivated by the probabilistic bounds given in [BS04] and provided in Table 3.1. Namely, the value of Γ^ϵ is such that there is a probability of $1 - \epsilon$ that the real cost will be no higher than the optimal solution cost whenever ψ is composed of independent and identically distributed random variables.

n	10%		20%		30%		40%		50%	
	DPA	MIP	DPA	MIP	DPA	MIP	DPA	MIP	DPA	MIP
50	0.051	18.6	0.078	15.2	0.091	10.2	0.094	7.67	0.115	7.56
100	0.358	70.3	0.519	52.1	0.537	29	0.634	23.5	0.674	21.5
200	2.77	1,600	3.72	940	3.95	417	4.69	326	5.23	183

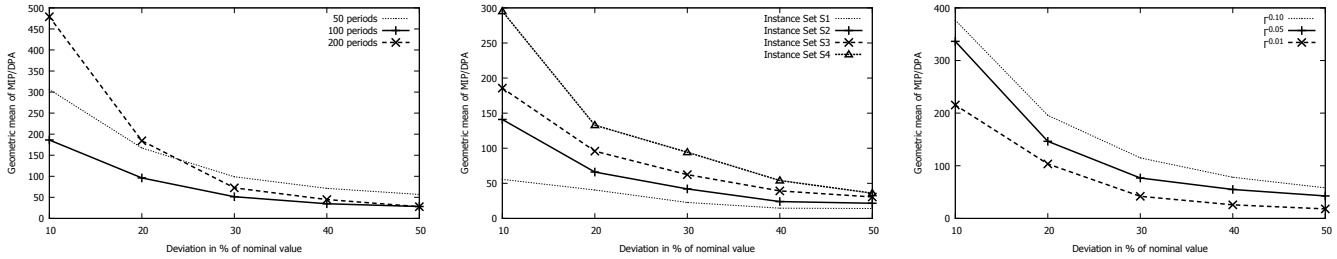
Table 3.2: Arithmetic means of the solution times.

Our experiments compare the DPA with the well-known MIP reformulation of (**AP**) (e.g. [BÖ08]) recalled below:

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{n} \varphi_i, \\
\text{s.t.} \quad & \varphi_i \leq \mu_i^0 + \mu_i^1 y_i + M_i u_i, \qquad \forall i = 1, \ldots, n, \\
& \varphi_i \leq \nu_i^0 + \nu_i^1 y_i + M_i (1 - u_i), \qquad \forall i = 1, \ldots, n, \\
& y_i = \sum_{i=1}^{i} \overline{\psi}_i + \widehat{\psi}_i \xi_i, \qquad \forall i = 1, \ldots, n, \\
& \sum_{i=1}^{n} \xi_i \leq \Gamma, \qquad \forall i = 1, \ldots, n, \\
& \xi \in \{0,1\}^n, \, u \in \{0,1\}^n, \, y \geq 0.
\end{aligned}
$$

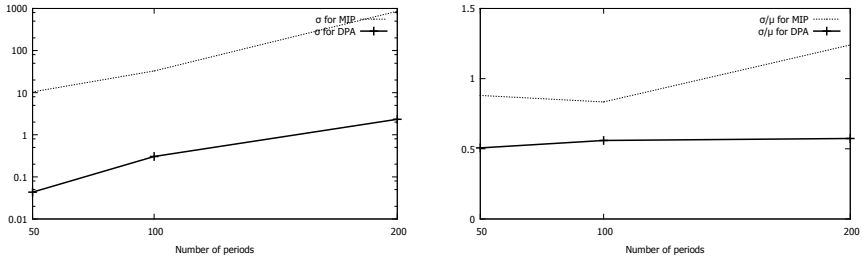
For each $i = 1, \ldots, n$, variables φ_i and y_i represent the value of functions $f_i(y_i(\psi))$ and $y_i(\psi)$, respectively. Then, for each $i = 1, \ldots, n$ variable u_i is equal to 0 iff $\mu_i(y_i(\psi))$ is larger than $\nu_i(y_i(\psi))$ and M_i is a large predefined value that cannot be smaller than $|f_i(y_i(\psi))|$ for every $\psi \in \Xi_\Gamma$.

The DPA was coded in C++ and compiled in a GNU G++ 4.5 compiler. The MIP formulation was implemented in C++ using Cplex Concert Technology 12.5 [CPL13]. The numerical experiments were carried out in an Intel(R) Core(TM) i7 CPU M60, 2.6Hz 4GB Ram machine.



(a) Varying the number of time periods. (b) Different sets of instances. (c) Different values of Γ.

Figure 3.1: Geometric mean of (time MIP)/(time DPA).



(a) Standard deviation represented in logarithmic scale. (b) Standard deviation divided by the arithmetic mean (μ).

Figure 3.2: Standard deviation of the solution times σ when varying the number of time periods.

3.6.2 Results

We provide in Figure 3.1 geometric means of the solution time of MIP divided by the solution time of DPA. The standard deviations of the solution times are illustrated on Figure 3.2 for both approaches. We present on Table 3.2 the arithmetic means of the solution times for the different number of time periods and levels of deviations.

Figure 3.1 shows that DPA clearly outperforms MIP, with means ranging up to 475 when $n = 200$ and the deviation is 10%. The different parameters strongly impact the respective solution times. First, we see on the charts from Figure 3.1 that, as expected from its theoretical complexity, higher levels of deviation slow down DPA. The number of time periods strongly affect both approaches, see Table 3.2. When the deviation is small, Figure 3.1a shows that the ratio between MIP and DPA increases with the value of n. In contrast, high levels of deviations tend to reduce the ratio between MIP and DPA. Figure 3.1b depicts the sensitivity of the ratio between the storage and backlog costs.

Our (unreported) results show that MIP is highly sensitive to the ratio, while DPA is not affected at all, explaining the results of Figure 3.1b. Finally, Figure 3.1c shows that higher values of Γ yields smaller ratios on average. Here, both approaches are strongly affected by the value of Γ, and the figure shows than DPA is more affected than MIP since the ratio decreases significantly when Γ rises.

The variations of the solution times are represented on Figure 3.2 for DPA and MIP, through the standard deviation (σ). Figure 3.2a presents these standard deviations in a logarithmic scale, which shows that the solution times of MIP vary between 2 and 3 order of magnitude more than the solution times of DPA. Figure 3.2b shows these standard deviations in a relative scale, dividing them by the associated arithmetic means. The figure shows that in a relative scale, DPA varies roughly twice as much as MIP. Our simple experiments show that DPA can be orders of magnitude faster than MIP, especially when the deviation level is low. Moreover, the absolute variability of MIP is much higher than the one of DPA, some instances being particularly hard to solve for the former. Notice also that we compared a simplistic implementation of DPA to the well-engineered MIP solver from CPLEX. It is likely that rules to eliminate dominated states would further improve our results, but this is out of the scope of the current chapter.

To conclude we intend to discuss some more general open questions. The dynamic programming algorithm presented in Chapter 3 can be easily parallelized, a naive strategy consists in, for each period i build a task to compute the recursive function for each value of Γ separately. This naive strategy has the downside that the tasks have a different computational load, a more egalitarian strategy consist in dividing, for a fixed period i and a fixed number of deviations Γ, the possible number of states for all the tasks. Notice that the egalitarian strategy has the same computational load for each task, but it task has little granularity, in this context, we can ask if the egalitarian strategy is even computationally worth for some size of problem, and if it is the case.

Chapter 4

Perfect information lower bounds for the adjustable problems

4.1 Introduction

Adjustable robust optimization is known for being $\mathcal{NP}$-hard, even in the case of a linear program with only two decision stages. In spite of its theoretical difficulty, the problem can be solved exactly by decomposition approaches whenever some assumptions hold [AP16, BCP14, ZZ11]. These approaches consider finite subsets of the uncertainty set and dynamically increase the number of considered scenarios by solving separation problems. The numerical tractability of the resulting algorithms highly depends on the complexity of the separation problem. For instance, the separation problem for the robust vehicle routing problem can be solved in polynomial time [ACF$^+$13], while the one related to facility location or network design problems require solving MILP with big-M coefficients [AP16, BCP14, ZZ11]. These decomposition approaches do not extend to multi-stage problems, because of the non-anticipativity constraints present in these problems. Stated simply, non-anticipativity constraints model the fact that optimization variables can only depend on past realizations of the uncertain parameters; they cannot adjust their decision to unknown realizations.

Given the difficulty of adjustable multi-stage robust problems, many researchers have developed heuristic approaches that try to provide feasible solutions for these problems. The bottom line of all these approaches is to restrict the set of feasible functions for the adjustable variables. The seminal paper in this line of research is [BGGN04] which restricts adjustable variables to affine functions of the uncertainties, which they call *affine decision rules*. Subsequent authors have studied more complex decision rules that offer more flexibility than affine decision rules while providing more or less tractable optimization problems.

In this chapter we provide a way to compute lower bounds for multistage robust optimization problems. Our approach relaxes the non-anticipativity constraints of the problem, thus yielding a relaxation of the original problem. This relaxation is well-known in the stochastic programming literature as the *perfect information* relaxation. We show in the chapter how the perfect information relaxation of the robust lot-sizing problems can be solved efficiently either through polynomial-time algorithms or MILP reformulations. Our experiments realized on lot-sizing instances inspired by the literature seem to indicate that the perfect information relaxation can be very tight. In this chapter, we pay particular attention to the budget uncertainty set.

4.2 The robust model

In this section we recall some definitions presented earlier in the manuscript and present some results and models related to robust lot-sizing problems. We consider an *uncertainty polytope* Ξ and we suppose that the demand at time period i is defined by the deviation function $d_i(\xi)$, which takes as argument an element of the uncertainty set Ξ and that such deviation function has an affine dependence of the uncertainty parameter, more clearly,

$$d_i(\xi) = \bar{d}_i + \sum_{j \in H} \hat{D}_{ij} \xi_j, \tag{4.1}$$

where $\bar{d}_i$ can be seen as the mean value of the clients demands for time period i and $\hat{D}$ is the *deviation matrix* (which can be estimated from historical data) that represents all temporal relations among demands. Equation (4.1) is sometime written shortly as $d_i(\xi) = \bar{d}_i + \hat{D}_i\xi$ where $\hat{D}_i$ represents the i-th row of matrix $\hat{D}$. We also assume that the demand functions are non-negative, more precisely $d_i(\xi) \geq 0$ for all $i \in H$ and $\xi \in \Xi$..

In the robust context, holding and backlog costs depend on the specific scenario ξ. Hence, they are represented by functions $s_i(\xi)$ and $r_i(\xi)$ for each time period i. The situation is more complex with production and setup costs. One could suppose that these features are independent of ξ, which would model the fact that all *decisions* must be taken at the beginning of the planning horizon, see for instance [ASNP16, BÖ08, BT06]. In this paper, we consider a more subtle approach where the productions and setups can be adjusted according to *past* demand realizations. Hence, these decisions are modeled by functions $x_i(\xi)$ and $y_i(\xi)$ for each time period i. Notice that, for each time period i, these functions must depend only on the demand revealed up to time period i. This is modeled by the *non-anticipativity* constraints

$$\begin{aligned} x_i(\xi) = x_i(\xi') \qquad & \forall \xi,\xi' \in \Xi, \mathrm{Proj}_{[1\ldots i]}(\xi) = \mathrm{Proj}_{[1\ldots i]}(\xi') \\ y_i(\xi) = y_i(\xi') \qquad & \forall \xi,\xi' \in \Xi, \mathrm{Proj}_{[1\ldots i]}(\xi) = \mathrm{Proj}_{[1\ldots i]}(\xi'). \end{aligned}$$

where $\mathrm{Proj}_{[1\ldots i]}(\xi)$ denotes the projection of ξ on its first i components. Said differently, the non-anticipativity constraints model the fact that x and y do not depend on future knowledge of the uncertainty. The mathematical formulation for the robust model follows.

$$((\mathbf{P}')) \qquad \min \qquad \kappa$$

$$s.t \qquad \kappa \geq \sum_{i \in H} (c_i x_i(\xi) + g_i y_i(\xi) + h_i s_i(\xi) + p_i r_i(\xi)) \qquad \forall \xi \in \Xi \tag{4.2}$$

$$s_{i+1}(\xi) = x_i(\xi) - d_i(\xi) + s_i(\xi) - r_{i-1}(\xi) + r_i(\xi) \qquad \forall i \in H, \forall \xi \in \Xi \tag{4.3}$$

$$x_i(\xi) \leq M y_i(\xi) \qquad \forall i \in H, \forall \xi \in \Xi \tag{4.4}$$

$$x_i(\xi) = x_i(\xi') \qquad \forall \xi, \xi' \in \Xi, \mathrm{Proj}_{[1\ldots i]}(\xi) = \mathrm{Proj}_{[1\ldots i]}(\xi') \quad \forall i \in H, \tag{4.5}$$

$$y_i(\xi) = y_i(\xi') \qquad \forall \xi, \xi' \in \Xi, \mathrm{Proj}_{[1\ldots i]}(\xi) = \mathrm{Proj}_{[1\ldots i]}(\xi') \quad \forall i \in H, \tag{4.6}$$

$$y(\xi) \in \{0,1\}^n, x(\xi), s(\xi), r(\xi) \geq 0 \qquad \forall \xi \in \Xi.$$

In this chapter, we will study bounding procedures for problem $\mathcal{BLSU}_{setup}(\Xi)$ as well as problems $\mathcal{BLSU}^{y}_{setup}(\Xi)$ and $\mathcal{BLSU}(\Xi)$. The first one considers that the setup decisions must be taken before knowing anything about the demand; that is, y becomes a vector of optimization variables that are independent of ξ. The second one looks at the problem without setup costs, which can be modeled by setting all components of y and g to 1 and 0, respectively. We denote these simplifications as $\mathcal{BLSU}^{y}_{setup}(\Xi)$ and $\mathcal{BLSU}(\Xi)$, respectively. Each of the tree models is relevant for specific applications. For instance, $\mathcal{BLSU}(\Xi)$ is close to the classical supply chain model addressed in most papers from the robust lot-sizing literature (e.g. [BT06, GdH13]). In contrast, models $\mathcal{BLSU}^{y}_{setup}(\Xi)$ and $(\mathbf{P}')$ are relevant for applications that involve fixed costs for the production due, for instance, to machine configurations.

In general, we assume that Ξ can be any non-empty polytope, described by the matrix

W with m rows and $|H| = n$ columns

$$\Xi = \{\xi \mid W\xi \leq q, \xi \geq 0\}. \tag{4.7}$$

In addition to general polytopes, we will also take a closer look at the complexity of the optimization problems when using the *budgeted uncertainty polytope* introduced in [BS04]. Given a positive real Γ, the budgeted polytope is defined as

$$\Xi_\Gamma = \left\{ \xi \mid \sum_{i \in H} |\xi_i| \leq \Gamma, \quad -1 \leq \xi_i \leq 1, \forall i \in H \right\}. \tag{4.8}$$

Strictly speaking, Ξ_Γ is not a special case of (4.7) because the uncertain parameters are allowed to take negative values in Ξ_Γ. This detail could be avoided by considering instead an extended formulation for Ξ_Γ, whith $\xi_i = \xi_i^+ - \xi_i^-$, ξ^+,$\xi^- \geq 0$ and doubling the number of rows of matrix $\hat{D}$. To simplify the exposure of the paper, we do not further comment on this and derive our results for the context specified above.

4.3 Bounds

We present in Subsections 4.3.1 and 4.3.2 approaches from the literature that provides upper and lower bounds for the optimal solution of the adjustable robust problems. Notice that these approaches cannot be applied to robust multi-stage optimization problems that contain adjustable integer variables, such as ($\mathbf{P}'$). Hence, in the following two sections, we assume that y does not depend on ξ, either because it is a nonadjustable vector of optimization variables (as in $\mathcal{BLSU}(\Xi)$) or because each of its components has been fixed to 1 (as in $\mathcal{BLSU}^y_{setup}(\Xi)$).

4.3.1 Affine decision rules in lot-sizing problems

The main idea of the approach is to impose that functions s_i,r_i and x_i depend affinely on ξ. Formally, these restrictions are modeled with constraints

$$x_i(\xi) = x_i^0 + \sum_{j=1}^{i} x_i^j \xi_j, \tag{4.9}$$

$$s_i(\xi) = s_i^0 + \sum_{j=1}^{n} s_i^j \xi_j, \tag{4.10}$$

$$r_i(\xi) = r_i^0 + \sum_{j=1}^{n} r_i^j \xi_j. \tag{4.11}$$

where x_i^0,s_i^0,r_i^0 and x_i^j,s_i^j,r_i^j for i,$j \in H$ are optimization variables. The right-hand side of (4.9) involves only the components of $\text{Proj}_{[1\ldots i]}(\xi) = \text{Proj}_{[1\ldots i]}(\xi)$. Hence, the equation models implicitly the non-anticipativity constraints introduced in the previous section.

Substituting $x_i(\xi)$,$s_i(\xi)$, and $r_i(\xi)$ in the rhs of (4.2)–(4.4) for each $i \in H$, we obtain an upper bound for $\mathcal{BLSU}(\Xi)$ and $\mathcal{BLSU}^y_{setup}(\Xi)$. Then, one can apply classical tools from robust optimization to the formulation to reformulate the upper bound as a compact linear program.

4.3.2 Dual affine decision rules

Recently, the authors of [KWG11] have proposed lower bounds for problems $\mathcal{BLSU}(\Xi)$ and $\mathcal{BLSU}^y_{setup}(\Xi)$, which they call *dual affine decision rules*. To be more precise, their approach is developed to provide lower bounds for multi-stage stochastic linear programs. To apply this technique to robust multi-stage programs, one needs to introduce artificial probability weights for the scenarios in Ξ. These probability weights are then used to formulate a lower bounding problem where the robust constraints are relaxed to expectation constraints. Then, a subtle reformulation allows them to provide a compact linear mixed integer formulation for the lower bounding problem. The reformulation is based on the use of convex duality and probability theory. One of the main difficulties of the method relies in the computation of the expectation matrix $\mathcal{M} = \mathbb{E}(\xi^T \xi)$. We redirect the interested reader to [KWG11] for full details.

4.3.3 Perfect information relaxation

The major impediment to the efficient solutions of problems $(\mathbf{P}')$, $\mathcal{BLSU}(\Xi)$ and $\mathcal{BLSU}^{y}_{setup}(\Xi)$ lies in the presence of the non-anticipativity constraints. Expressing non-anticipativity constraints is not easy in general and strongly depends on the particular structure of the considered set Ξ. In what follows, we propose a lower bounding problem for $(\mathbf{P}')$ that relaxes the non-anticipativity constraints from $(\mathbf{P}')$, which we call the *problem with perfect information*. Unlike the affine decision rules and the dual version presented in the previous sections, the perfect information relaxation can be applied to $(\mathbf{P}')$ regardless to the dependency of y on ξ. The associated optimization problems are denoted by $\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)$, $\mathcal{PI}-\mathcal{BLSU}^{y}_{setup}(\Xi)$, and $\mathcal{PI}-\mathcal{BLSU}(\Xi)$, for $(\mathbf{P}')$, $\mathcal{BLSU}^{y}_{setup}(\Xi)$ and $\mathcal{BLSU}(\Xi)$ respectively.

$$
\begin{aligned}
(\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)) \quad \min \quad & \kappa & & \\
s.t \quad & \kappa \geq \sum_{i\in H}(c_i x_i(\xi)+g_i y_i(\xi)+h_i s_i(\xi)+p_i r_i(\xi)) & & \forall \xi\in\Xi \\
& s_{i+1}(\xi)=x_i(\xi)-d_i(\xi)+s_i(\xi)-r_{i-1}(\xi)+r_i(\xi) & & \forall i\in H, \forall \xi\in\Xi \\
& x_i(\xi)\leq y_i(\xi)M & & \forall i\in H, \forall \xi\in\Xi \\
& y(\xi)\in\{0,1\}^n, x(\xi), s(\xi), r(\xi)\geq 0 & & \forall \xi\in\Xi.
\end{aligned}
$$

Hence, a formulation for problem $\mathcal{PI}-\mathcal{BLSU}^{y}_{setup}(\Xi)$ can be obtained from the above formulation by removing the dependency of ξ from y, while a formulation for $\mathcal{PI}-\mathcal{BLSU}(\Xi)$ is obtained by removing the variables y and constraints associated with it. This approach is well-known in stochastic optimization to examine the quality of proposed solutions. In particular, it is used to compute the so-called *expected value of perfect information* which defines the maximum price that one would be ready to pay to obtain perfect information about the actual scenario, see [BL11].

4.4 Solving the problem with perfect information

Let us first introduce some useful definitions. The *cumulative cost* w_{ij} represents the unitary cost of producing at time period i to satisfy the demand of time period j:

$$w_{ij} = c_i + \sum_{l=i}^{j-1} h_l + \sum_{l=j}^{i-1} p_l.$$

Notice that the above is well-defined, since for a fixed period i, $i \neq j$, only one of the two summations is not empty. For the sake of simplicity, we denote by ω_j the *minimum cumulative cost* for period j, the smallest among values $\{w_{ij}, i \in H\}$. Finally, we denote by $opt(X)$ the optimal solution cost of any optimization problem X.

In the following, we discuss how to solve the optimization problems obtained by relaxing the non-anticipativity constraints. We first focus on problem $\mathcal{PI} - \mathcal{BLSU}(\Xi)$, then we address problem $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)$, and we finish with problem $\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)$. For each problem, we present a generic solution algorithm that can handle general uncertainty poltyopes and more efficient algorithms that are tailored for the budgeted uncertainty polytope.

4.4.1 No setup

We first deal with the robust problem without setup costs $\mathcal{PI} - \mathcal{BLSU}(\Xi)$, which can be formulated as follows

$$
\begin{array}{lll}
(\mathcal{PI} - \mathcal{BLSU}(\Xi)) \quad \min & \kappa & \\
s.t & \kappa \geq \sum_{i \in H} (c_i x_i(\xi) + h_i s_i(\xi) + p_i r_i(\xi)) & \forall \xi \in \Xi \\
& s_{i+1}(\xi) = x_i(\xi) - d_i(\xi) + s_i(\xi) - r_{i-1}(\xi) + r_i(\xi) & \forall i \in H, \forall \xi \in \Xi \\
& x(\xi), s(\xi), r(\xi) \geq 0 & \forall i \in H, \forall \xi \in \Xi.
\end{array}
$$

We show that $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ is equivalent to problem

$$\max_{\xi \in \Xi} \sum_{i \in H} \omega_i d_i(\xi),$$

where ω_i is the minimum cumulative cost of period i as defined previously. Hence, the result shows that the complexity of $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ is related to the complexity of optimizing an affine function over Ξ.

Theorem 1. *Let Let Ξ be any uncertainty set. Then,*

$$opt(\mathcal{PI} - \mathcal{BLSU}(\Xi)) = \max_{\xi \in \Xi} \sum_{i \in H} \omega_i d_i(\xi)$$

Proof. Because there is no capacity constraint, one readily verifies the following. For each time period i, there exists a unique time period j for which all demand of i is produced, which corresponds to the period that provides the minimum cumulative cost to serve the demand of time period i. Hence, given any $\xi \in \Xi$, we have to pay the total cost

$$\sum_{i \in H} \omega_i d_i(\xi)$$

Then, the absence of setup costs implies that the time period yielding the minimum production cost does not depend on ξ, proving the result. □

Theorem 1 implies that $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ is polynomially solvable since linear programming is polynomially solvable. We show in the next result that we can get faster algorithms for Ξ when Γ is integer.

Corollary 2. *Let $\Xi = \Xi^{\Gamma}$ and define the subset $\mathbf{\Gamma}(\omega^T \hat{D}) \subseteq H$ that contains the indices of the Γ largest elements of the vector $(|\sum_{i \in H} \omega_i \hat{D}_{ij}|, j \in H)$, and $\mathbf{\Gamma'}(\omega^T \hat{D})$ that denotes*

the (Γ+1)-largest element of that vector. The following holds:

$$opt(\mathcal{PI} - \mathcal{BLSU}(\Xi)) = \sum_{i \in H} \omega_i \bar{d}_i + \sum_{i \in \mathbf{\Gamma}(\omega^T \hat{D})} |\omega^T \hat{D}_i| + (\Gamma - \lfloor \Gamma \rfloor)|\omega^T \hat{D}_{\mathbf{\Gamma}'(\omega^T \hat{D})}|.$$

Moreover, $opt(\mathcal{PI} - \mathcal{BLSU}(\Xi))$ can be computed in $O(n^2)$.

Proof. We obtain immediately from Theorem 1 that

$$\begin{aligned}
opt(\mathcal{PI} - \mathcal{BLSU}(\Xi)) &= \max_{\xi \in \Xi_\Gamma} \sum_{i \in H} \omega_i d_i(\xi) \\
&= \max_{\xi \in \Xi_\Gamma} \sum_{i \in H} \left(\omega_i \bar{d}_i + \sum_{j \in H} \omega_i \hat{D}_{ij} \xi_j \right) \\
&= \sum_{i \in H} \omega_i \bar{d}_i + \max_{\xi \in \Xi_\Gamma} \sum_{i,j \in H} \omega_i \hat{D}_{ij} \xi_j \\
&= \sum_{i \in H} \omega_i \bar{d}_i + \max_{\substack{\sum_{i \in H} |\xi_i| \leq \Gamma \\ -1 \leq \xi_i \leq 1}} \sum_{i,j \in H} \omega_i \hat{D}_{ij} \xi_j \\
&= \sum_{i \in H} \omega_i \bar{d}_i + \sum_{i \in \mathbf{\Gamma}(\omega^T \hat{D})} |\omega^T \hat{D}_i| + (\Gamma - \lfloor \Gamma \rfloor)|\omega^T \hat{D}_{\mathbf{\Gamma}'(\omega^T \hat{D})}|
\end{aligned}$$

Regarding the complexity, notice that we must calculate all cumulative costs, which takes $O(n^2)$. Then, we must calculate the minimum cumulative cost for every period, which takes $O(n^2)$. Finally, we must compute the vector ($|\omega^T \hat{D}_i|$,$i \in H$), which takes $O(n^2)$ and chooses the Γ larger values of it, which takes $O(\Gamma \log n)$. Hence, the complexity of this strategy is $O(3n^2 + \Gamma \log n) = O(n^2)$. □

4.4.2 Non-adjustable setup

In this section, we address problem $\mathcal{PI} - \mathcal{BLSU}^y_{setup}(\Xi)$, which can be formulated as

$$
\begin{aligned}
(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)) \quad \min \quad & \kappa & \\
s.t \quad & \kappa \geq \sum_{i \in H} (c_i x_i(\xi) + g_i y_i + h_i s_i(\xi) + p_i r_i(\xi)) & \forall \xi \in \Xi \\
& s_{i+1}(\xi) = x_i(\xi) - d_i(\xi) + s_i(\xi) - r_{i-1}(\xi) + r_i(\xi) & \forall i \in H, \forall \xi \in \Xi \\
& x_i(\xi) \leq y_i M & \forall i \in H, \forall \xi \in \Xi \\
& x(\xi), s(\xi), r(\xi) \geq 0 & \forall \xi \in \Xi \\
& y \in \{0,1\}^n.
\end{aligned}
$$

We address in this section $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)$ for general polytopes and show how the problem can be reformulated as mixed-integer linear program.

Theorem 2. *Let Ξ be the uncertainty polytope defined in* (4.7)*. The problem $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)$ can be solved by the following mixed integer linear program*

$$
\begin{aligned}
opt(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)) = \min \ & \sum_{l=1}^{m} q_l z_l + \sum_{i,j \in H} v_{ij} w_{ij} \bar{d}_j + \sum_{j \in H} g_j y_j & \\
s.t \quad & \sum_{l=1}^{m} W_{lk} z_i \geq \sum_{i,j \in H} v_{ij} w_{ij} \hat{D}_{jk} & \forall k \in H, \\
& \sum_{i \in H} v_{ij} = 1 & \forall j \in H \\
& v_{ij} \leq y_i & \forall i,j \in H \\
& v \in \{0,1\}^{n^2}, y \in \{0,1\}^n, z \geq 0,
\end{aligned}
$$

Proof. Consider a fixed binary vector y for problem $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)$. We let $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)$ be problem $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)$ with the values of the setup fixed according to vector y.
It is clear that

$$
opt(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)) = \min_{y \in \{0,1\}^n} opt(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)). \tag{4.12}
$$

Problem $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)$ is a problem without setup costs, that is, it is a special case of $\mathcal{PI} - \mathcal{BLSU}(\Xi)$. Let us define

$$\omega_j(y) = \min \left\{ \sum_{i \in H} v_{ij} w_{ij} \text{ s.t. } \sum_{i \in H} v_{ij} = 1,\ v_{ij} \leq y_i, \forall i \in H\, v \in \{0,1\}^{n^2} \right\} \tag{4.13}$$

as the smallest cumulative cost for period j, among all costs w_{ij} such that $y_i = 1$. Applying Theorem 1 to $\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)$, we obtain that

$$opt(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)) = g_j y_j + \max_{\xi \in \Xi} \sum_{j \in H} \omega_j(y) d_j(\xi) = \max_{\xi \in \Xi} \sum_{j \in H} \omega_j(y) \overline{d}_j + \sum_{j,k \in H} \omega_j(y) \hat{D}_{jk} \xi_k + \sum_{j \in H} g_j y_j. \tag{4.14}$$

Since Ξ is non-empty and bounded, we can apply the linear programming strong duality to the maximization problem defined over $\Xi = \{\xi \mid W\xi \leq q, \xi \geq 0\}$ in the rhs of (4.14). Introducing the non-negative dual variables $\{z_l, l = 1, \ldots, m\}$ for the constraints defining Ξ, (4.14) becomes

$$\begin{aligned} opt(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)) = \min \quad & \sum_{l=1}^{m} q_l z_l + \sum_{j \in H} \omega_j(y) \overline{d}_j + \sum_{j \in H} g_j y_j \\ s.t \quad & \sum_{l=1}^{m} W_{lk} z_l \geq \sum_{j \in H} \omega_j(y) \hat{D}_{jk} \quad \forall k \in H, \\ & z \geq 0. \end{aligned}$$

Plugging the definition (4.13) of $\omega_j(y)$ into the above formulation and removing the minimization over v from the constraints, we obtain

$$
\begin{aligned}
opt(\mathcal{PI} - \mathcal{BLSU}^{y}_{setup}(\Xi)(y)) = \min \; & \sum_{l=1}^{m} q_l z_l + \sum_{i,j \in H} v_{ij} w_{ij} \bar{d}_j + \sum_{j \in H} g_j y_j \\
s.t \quad & \sum_{l=1}^{m} W_{lk} z_i \geq \sum_{i,j \in H} v_{ij} w_{ij} \hat{D}_{jk} && \forall k \in H, \\
& \sum_{i \in H} v_{ij} = 1 && \forall j \in H \\
& v_{ij} \leq y_i && \forall i,j \in H \\
& v \in \{0,1\}^{n^2}, z \geq 0,
\end{aligned}
$$

and the result follows from (4.12). □

4.4.3 Adjustable setup

In this section we present a linear program that computes the value of $opt(\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi))$ for general polytopes Ξ. Similarly to Theorem 1, we show that the complexity of problem $\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)$ is related to the complexity of optimizing an affine function over Ξ. This time however, the linear program to be solved contains $O(n^3)$ constraints in addition to those describing Ξ. Let us first recall a well-known dynamic programming algorithm to solve the deterministic problem $\mathcal{P}$.

Lemma 9 (Dynamic Program [PW06])**.** *Problem $\mathcal{P}$ can be solved by the following recursive function*

$$opt(\mathcal{P}) = G(n)$$

where

$$G(j) = \min_{\{i,k | i<k\leq j\}} \left(G(i) + g_k + \sum_{l=i+1}^{j} w_{kl} d_l \right), \tag{4.15}$$

where $G_0 = 0$.

To compute $\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)$, it will be useful to reformulate the above dynamic program as the linear program provided in the next lemma.

Lemma 10. *The optimal solution cost of $\mathcal{P}$ is equal to the optimal solution cost of the following linear program, with optimization variables u_i for each $i \in H$*

$$\begin{aligned} \max \quad & u_n \\ s.t \quad & u_j \leq u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l \qquad \forall i < k \leq j, i \in H \cup \{0\}, j \in H. \\ & u_0 = 0 \end{aligned}$$

Proof. We introduce optimization variable u_i to represent the value of $G(i)$. The counterpart of equation (4.15) for u is

$$u_j = \min_{\{i,k|i \leq k \leq j\}} \left(u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l \right). \tag{4.16}$$

Then, we relax the equality in (4.16) to

$$u_j \leq \min_{\{i,k|i \leq k \leq j\}} \left(u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l \right), \tag{4.17}$$

and ensure that the objective function of the linear program maximizes the value of u_j. Constraint (4.17) can be easily linearized to

$$u_j \leq u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l \quad \forall i \leq k \leq j, i \in H. \tag{4.18}$$

Combining linear constraints (4.18) with the objective function that maximizes u_n yields the result. □

Using the above results, we propose a linear programming reformulation for $\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)$.

Theorem 3. *For any uncertainty set Ξ, problem $\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)$ can be solved by*

the following linear program in optimization vectors u and ξ

$$\begin{aligned}\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi) = \max \quad & u_n \\ s.t \quad & u_j \leq u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l(\xi) \qquad \forall i \leq k \leq j, i \in H, j \in H \\ & \xi \in \Xi\end{aligned}$$

Proof. We define $\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi)$, as the problem $\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)$ restricted to a fixed element $\xi \in \Xi$. One readily sees that

$$opt(\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)) = \max_{\xi\in\Xi} opt(\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi))$$

and moreover, $\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi)$ is a deterministic lot-sizing problem. Hence, we can apply Lemma 9 and use the following dynamic program for solving problem $\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi)$

$$opt(\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi)) = G_\xi(n)$$

where

$$G_\xi(j) = \min_{\{i,k|i\leq k\leq j\}} \left(G_\xi(i) + g_k + \sum_{l=i}^{j} w_{kl} d_l(\xi) \right).$$

Using Lemma 10, $\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi)$ has the same optimal solution as the linear program

$$\begin{aligned}opt(\mathcal{PI}-\mathcal{BLSU}_{setup}(\Xi)(\xi)) = \max \quad & u_n \\ s.t \quad & u_j \leq u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l(\xi) \qquad \forall i \leq k \leq j, i \in H, j \in H.\end{aligned} \tag{4.19}$$

Now, as $opt(\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)) = \max_{\xi \in \Xi} opt(\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)(\xi))$ we have that

$$
\begin{aligned}
opt(\mathcal{PI} - \mathcal{BLSU}_{setup}(\Xi)) = \max \quad & u_n \\
s.t \quad & u_j \leq u_i + g_k + \sum_{l=i}^{j} w_{kl} d_l(\xi) \qquad \forall i \leq k \leq j, i \in H, j \in H \\
& \xi \in \Xi
\end{aligned}
$$

Recalling that $d(\xi)$ is an affine function of ξ ends the proof. □

4.5 Numerical experiments

In this section, we present the experimental results obtained with the relaxation presented in the last sections for problems $\mathcal{BLSU}^{y}_{setup}(\Xi)$ and $\mathcal{BLSU}(\Xi)$. We use the budgeted polytope Ξ_Γ as uncertainty set due to its importance in the literature and the identity as the deviation matrix, so that the demand can be expressed as $d(\xi)_i = \bar{d}_i + \hat{d}_i \xi_i$.

We compare the quality of our lower bound with the one provided by dual affine decision rules from [KWG11], recalled in Section 4.3.2. We also compare these lower bounds with the upper bound provided by the affine decisions rules (called primal affine decision rules [KWG11]), recalled in 4.3.1. We do not carry out experiments for problem ($\mathbf{P}'$) because the primal and dual affine decision rules methods cannot be applied to this problem so that we have no comparison possible for our method.

Concerning the probabilistic distribution used in dual affine decision rules, we use a uniform distribution over the extremes points of the uncertainty polytope. The reasons for that are two-fold: first we need a distribution that allows us to calculate simply matrix $\mathcal{M}$ (the expectation matrix); second, we do not have a dominating element among the extreme points of the considered polytope, contrasting with the problem studied in [KWG11].

The tests were carried out on an Intel(R) Core(TM) i7 CPU M60, 2.6Hz 4GB Ram machine and all formulations and algorithms were coded in C++, compiled with a GNU

G++ 4.5 compiler and IBM CPLEX 12.3. In the next subsection, we explain how the instances are built and which experiments are carried out.

4.5.1 Instances

We start with the description of the instances. We consider two sets of instances: `DYN` and `DOWN`. The instances in the set `DYN` represent lot-sizing problems in which the costs associated are seasonal. The instances of that set are inspired by [BGGN04] and they fulfill a criteria known as *Wagner-Within*, which has been introduced in [PW94]. Roughly speaking, this criteria implies that it is always cheaper to produce the client demands of each period at the period itself. More precisely, we have for each period i that $c_i = 20 + 5\sin(\frac{i\pi}{12})$, $h_i = 5 + 2\sin(\frac{i\pi}{12})$ and $p_i = 7 + 2\sin(\frac{i\pi}{12})$.

To contrast with that set of instances, we use a second set of instances that do not fulfill the Wagner-Within criteria. In the set of instances named `DOWN`, for each each period i is cheaper to produce the client demand at period $3\lfloor\frac{i}{3}\rfloor$. More precisely, we have that $c_i = 10 + 5(i \mod 3)$, $h_i = 3$ and $p_i = 4$.

We consider the deviations fixed as 20% of the nominal demand, and the nominal deviations are given by the formula $100 + 50\sin(\frac{i\pi}{12})$ for each period i. We consider horizons of planning that have 20, 30, 40, 50, 60, 70, 80, 90 and 100 periods. We tested two different values of the parameter Γ for each number of periods in the horizon of planning, each of them inspired by the probabilistic bounds computed in [BS04], see also [Pos14]. Table 4.1 presents the values used for the experiments.

4.5.2 Lot-Sizing problem without setup costs

We report below a comparison of the two lower bounds and as well as their solution times. Let $AFFINE$ stand for the affine decision rules presented in the Section 4.3.1, PI for the perfect information proposed and $DUAL$ for the dual affine decision rules presented in Section 4.3.2. We compare the *optimality gaps* of PI and $DUAL$ using the solution

H	$\Gamma^{0.01}$	$\Gamma^{0.1}$
10	5	8
20	7	11
30	8	14
40	9	16
50	10	17
60	11	19
70	12	20
80	12	22
90	13	23
100	14	24

Table 4.1: Values of parameter Γ inspired by the probabilistic bounds from [BS04].

of the approach $AFFINE$ as upper bound. Specifically, we define the *approximative optimality gap* for PI for the instance I as

$$\frac{opt(AFFINE(I)) - opt(PI(I))}{opt(AFFINE(I))}.$$

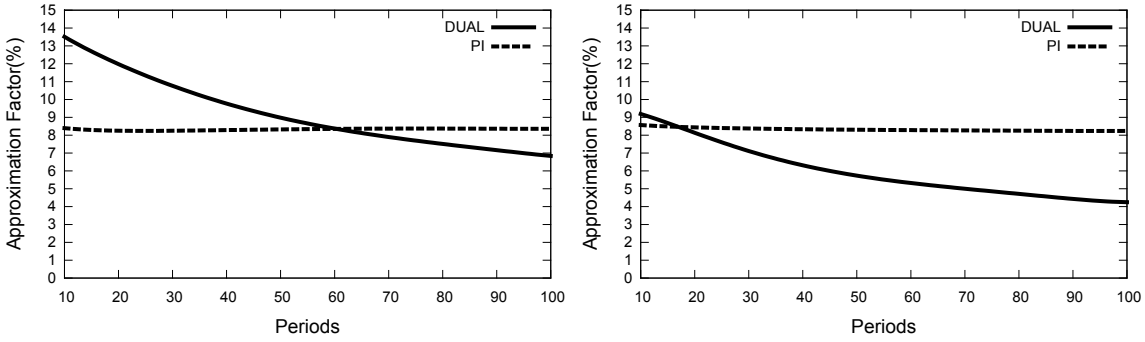
We define similarly the approximative approximation gap for the problem $DUAL$ concerning the instance I as

$$\frac{opt(AFFINE(I)) - opt(DUAL(I))}{opt(AFFINE(I))}.$$

In the following, we report the approximative optimality gap and the solution times, computed for each number of periods and each set of instances.

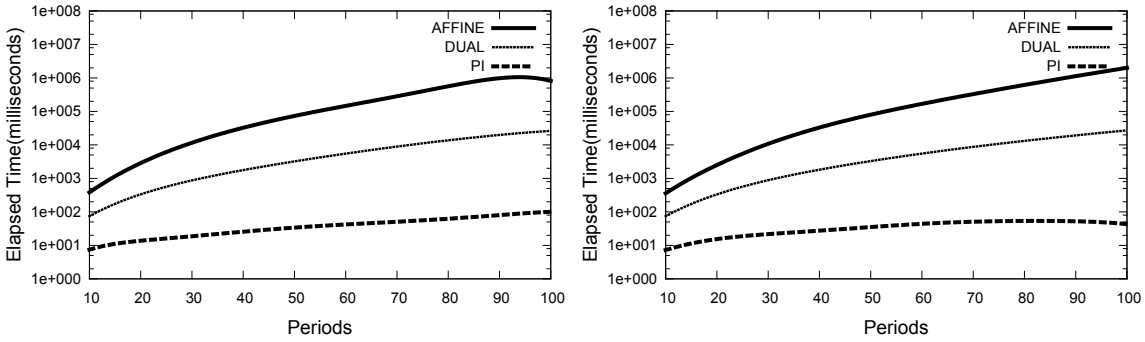
Instances `DOWN`

The approximation obtained by PI is nearly constant at 8% while the one of $DUAL$ is slightly decreasing with the number of periods, see Figure 4.1. In any case, approach PI still provides a reliable approximation with a computation time much smaller than the one required by $DUAL$ as presented in Figure 4.2



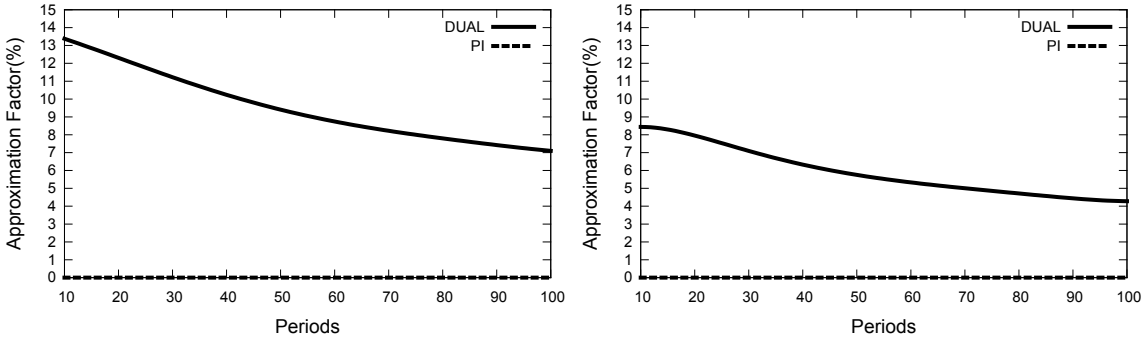
(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

Figure 4.1: The optimality gaps for $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DOWN.



(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

Figure 4.2: Elapsed time of $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DOWN.

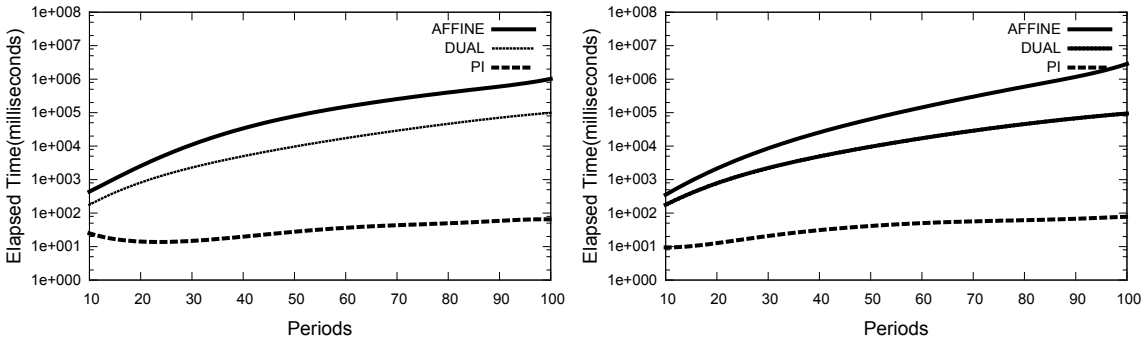


(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

Figure 4.3: The optimality gaps for $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DYN.

Instances DYN

The bound provided by PI can prove the optimality of the affine decisions rules while the dual affine decision rules decrease to reach a gap of 5%, see in Figure 4.3. It seems



(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

Figure 4.4: Elapsed time of $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DYN.

that the Wagner-Within criteria, where it is also always cheaper to produce the demand of some period i at the period itself, strongly reduces the impact of the non-anticipativity constraints. As for the computation time, our approach is much faster than the other two, as reported in Figure 4.4. One can also notice that the dual affine decision rules are more time consuming for the instance set DYN than they are for the instance set DOWN.

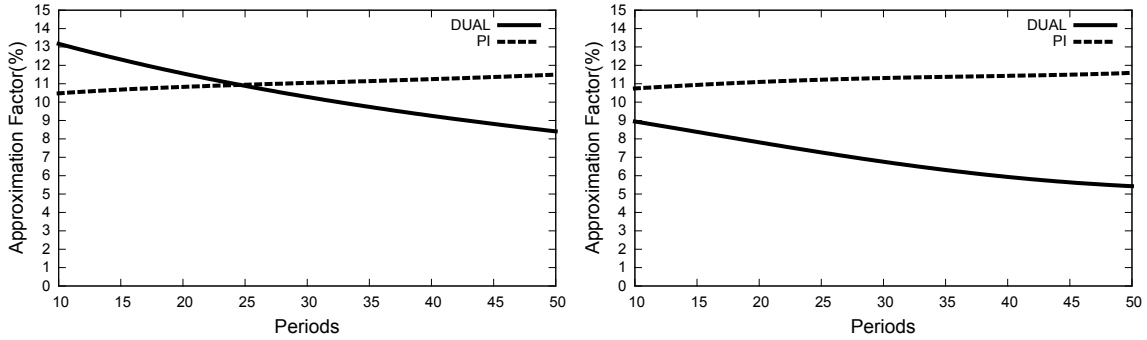
4.5.3 Lot-Sizing problem with setup costs

As before, $AFFINE$ stands for the affine decision rules presented in the Section 4.3.1, PI for the perfect information proposed in this paper and $DUAL$ for the dual affine decision rules presented in Section 4.3.2. We compare the *optimality gaps* of PI and $DUAL$ using the solution cost of $AFFINE$ as upper bound.

In the following, we report the approximative optimality gap and the solution times, computed for each number of periods and each set of instances. Two remarks must be done at this point. First, we solve the problem PI with the MILP proposed in Corollary 2.

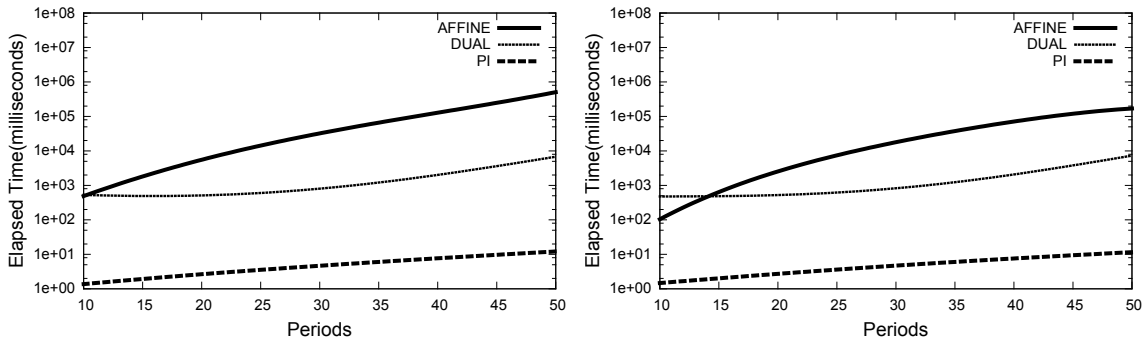
Instances DOWN

The approximation obtained by PI is nearly constant at 11% while the one of $DUAL$ is decreasing with the number of periods, see Figure 4.5. Figure 4.6 then shows that PI can be solved much faster than $DUAL$.



(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

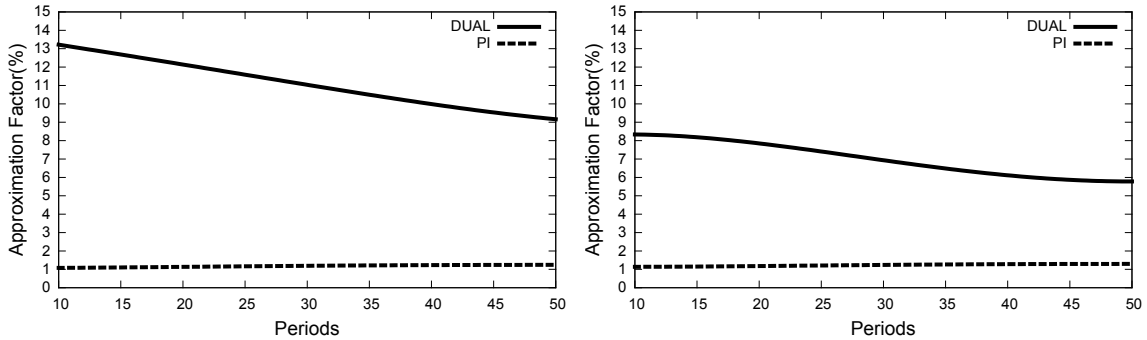
Figure 4.5: The optimality gaps for $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DOWN.



(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

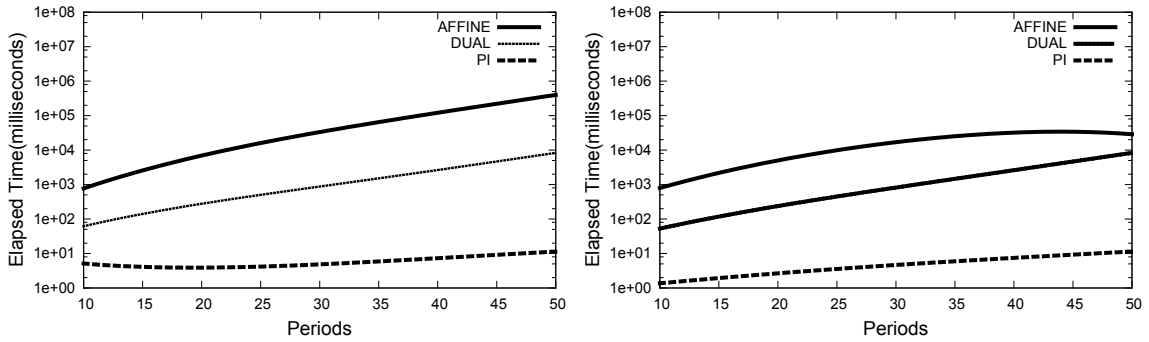
Figure 4.6: Elapsed time of $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DOWN.

Instances DYN



(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

Figure 4.7: The optimality gaps for $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DYN.



(a) $\Gamma^{0.1}$ over DOWN. (b) $\Gamma^{0.01}$ over DOWN.

Figure 4.8: Elapsed time of $\mathcal{PI} - \mathcal{BLSU}(\Xi)$ for the instances in DYN.

Although we are not able to prove the optimality of the affine decision rules, the lower bound provided by PI is really close to the solution cost provided by $AFFINE$ (roughly 1%). And again, concerning the elapsed time, the proposed method is 2, some times 3 magnitudes orders faster than the dual affine decision rules. Again, Figure 4.8 shows PI to be much faster than $DUAL$.

4.6 Concluding remarks

We have adapted in this paper the perfect information relaxation, well-known in the stochastic programming literature, to the robust lot-sizing problem, yielding a lower bound for the robust problem. We could prove that computing the bound can be done in polynomial time for several variants of the problem, using dedicated combinatorial algorithms or linear programs. Our numerical results, realized on instances inspired by the literature, suggest that the new lower bound can be tight and can be solved much faster than the bounds based on decision rules. In addition, our approach can handle binary adjustable variables, which is not the case of primal and dual affine decision rules.

The tractability of our lower bound is highly problem-dependent. Hence, it would be interesting to investigate its tractability for other class of multi-stage robust optimization problems and other uncertainty sets. Concerning the extensions of the problems under

study, we think it may be interesting to consider problems with capacities on the production and the storage. An alternative extension would consider problems with multiple items. Concerning the uncertainty set, it could be interesting to look for dedicated algorithms for ellipsoidal uncertainty sets, which have recently been the topic of several papers in the combinatorial robust optimization literature (e.g. [BBI14]).

Another interesting aspect of the perfect information relaxation is that it turns multi-stage problems into two-stages problems, for which several papers have recently proposed exact solution algorithms based on variants of the Benders decomposition (e.g. [AP16, BCP14, ZZ11]). Hence, one could use these algorithms to solve the relaxation for much more general problems than the variants of the lot-sizing considered herein, and assess the quality of the obtained lower bound.

Part III

Other Contribution

This part contains the second set of contributions. The main contribution of this part concerns the sensor location problem, a special case of facility location problem. We manage to provide a linear formulation of different variety of this problem and especially provide a full study of the equitable robust sensor location problem.

Chapter 5

Proportional and maxmin fairness for the sensor location problem with chance constraints

5.1 Introduction

In this chapter, we consider the problem of installing facilities at strategic locations in order to monitor and protect numerous important locations. Each location can be simultaneously protected by multiple facilities. Concrete examples include airports where various locations, such as terminals, baggage areas, control towers, gates, runways, etc. must be protected. Other examples include shopping malls and entertainment parks (e.g., Disney World) where large numbers of people assemble at many locations, and strategic complexes such as hospitals, power plants, and military installations. The objective is to provide (i) equitable protection to all locations when the number of sensors that can be placed is limited (ii) robust protection under possibly full or partial failures of sensors. The Equitable Sensor Location Problem [IK88, Lus12, LNPS16] is an extension of the Equitable Facility Location Problem, see Ogryczak [Ogr97]. The latter considers placing facilities, such as police stations, emergency rooms, etc., so as to provide equitable service

to all neighborhoods, where people in each of the neighborhoods are served by the closest facility. Versions of this problem are proposed in Neidhardt, Luss, and Krishnan [NLK08], and in Luss [Lus12] [Section 7.2.2].

In this study we explore models with two different objective functions often used to model optimization problems with fairness criteria: (i) Lexicographic maximin (or minimax) optimization and (ii) Proportional fairness optimization. The contribution of this work stands in providing a full study on the different variants of the probabilistic equitable sensor location problem. The contribution is twofold: (i) modeling and solving the probabilistic equitable sensor location problem and the resilient variant; (ii) modeling and solving the ambiguous equitable sensor location problem. Note that the resilient variant of the equitable sensor location stands for the case when sensors are subject to failures while the ambiguous equitable sensor location problem considers the case with uncertain probabilities. The paper is organized as follows. Section 2 gives preliminaries on equity and proportional fairness together with a new result. Section 3 studies the basic equitable sensor location problem and proposes an integer linear program. The same trick is used to compute the proportionally fair solution in linear time. Next, we present a similar model for the equitable resilient sensor location problem, assuming that sensors can fail in some situation. Section 4 is devoted to the ambiguous sensor location problem, where probabilities are assumed to vary in a finite set. We prove that the the proportional fairness version of the ambiguous problem is $\mathcal{NP}$-hard in the strong sense and present mixed-integer linear programming formulations. Section 5 presents numerical results. The numerical results show the value of the resilient and ambiguous solutions when compared to the deterministic ones.

5.2 Preliminaries

This section is devoted to preliminaries on equity and proportional fairness together with a new result which will be very useful in writing down the linear integer model for the basic

sensor location problem. Next, we deduce the mathematical formulation of the problem and discuss solution methods for both equitable and proportionally fair variants.

Let us start by introducing formally the notion of equity as discussed in this paper. The equity notion is closely related to lexicographic optimization as remarked in numerous studies, [BGH92, LB05, NP08, Lus12, OLP+14] etc. We recall some definitions on lexicographic ordering, useful for a better understanding of the study. A vector γ is lexicographically greater (resp. lower) than γ' if there exists $s \in \{1, ..., n\}$ such that $\gamma_p = \gamma'_p$, for all $p \in \{1, ..., s-1\}$ and $\gamma_s > \gamma'_s$ (resp. $\gamma_s < \gamma'_s$). A vector γ is lexicographically maximal (resp. minimal) in X if for every vector $\gamma' \in X$, γ is lexicographically greater (resp. lower) than or equal to γ'.

Let $\overrightarrow{\gamma}$ (resp. $\overleftarrow{\gamma}$) be the vector γ with its indices reordered so that the components are in non-decreasing (resp. non-increasing) order. A feasible vector is defined as leximin maximal [LB05] as follows: A vector $\gamma \in X$ is leximin maximal if for every vector $\gamma' \in X$, $\overrightarrow{\gamma}$ is lexicographically greater than or equal to $\overrightarrow{\gamma'}$. Similarly, one can define leximax minimality as follows: a vector $\gamma \in X$ is leximax minimal if for every vector $\gamma' \in X$, $\overleftarrow{\gamma}$ is lexicographically lower than or equal to $\overleftarrow{\gamma'}$.

Let us look now at the solution methodology. We define $\Gamma \subset \mathbb{R}^m$ as the set of vectors γ for which the following set is non-empty:

$$\{f_i(x) \geq \gamma_i; \quad i \in 1, ..., m, x \geq 0, x \in \mathbb{R}^n\}. \tag{5.1}$$

We say that γ is feasible if $\gamma \in \Gamma$. Let us focus on the feasible vectors γ that are leximin maximal. Computing a leximin maximal vector for the inequation's system (5.1) when $f_i(x)$ are linear is relatively easy as shown in [NO07]. Then, one can compute a leximin maximal vector among the feasible vectors by solving a sequence of at most m linear programs. At iteration i one computes the highest value that can take the i^{th} smaller component of the solution vector.

Similar results can be drawn for the following system of functions:

$$\{f_i(x) \leq \gamma_i; \quad i \in 1, ..., m, x \geq 0, x \in \mathbb{R}^n\}, \tag{5.2}$$

where we look for a feasible vector γ which is leximax minimal.

The above results are shown for the systems where $x \in \mathbb{R}^n$. Nevertheless, an effective iterative method based on OWA (Ordered Weighted Average) criteria developed in [KOW04] allows to solve optimally the problem for the discrete case (i.e. $x \in \mathbb{Z}_+^n$). Generally speaking, the authors make the connection between the notion of equitably efficient solutions and the Pareto-optimality concept by proposing some special types of aggregations, especially the Ordered Weighted Averaging aggregations initially proposed in [Yag88]. The principle of the method is to reformulate the LP problem with an objective function involving the cumulated OWA and with some extra linear constraints and integer variables. This formulation is valid for non-convex (discrete) feasible sets Γ, where the first approach is not applicable.

Let us consider some strictly increasing function ϕ and the system composed of functions $\phi \circ f_i$. Recall that the operator $\circ$ stands for the function composition operator. It can be shown that the following result holds.

Proposition 4. *Let ϕ be a strictly increasing function in $\mathbb{R}$. A vector γ feasible for* (5.1) *is leximin maximal if and only if the vector $(\phi(\gamma_1), \ldots, \phi(\gamma_m))$ is leximin maximal for the corresponding system composed of functions $\{\phi \circ f_i, i \in M\}$.*

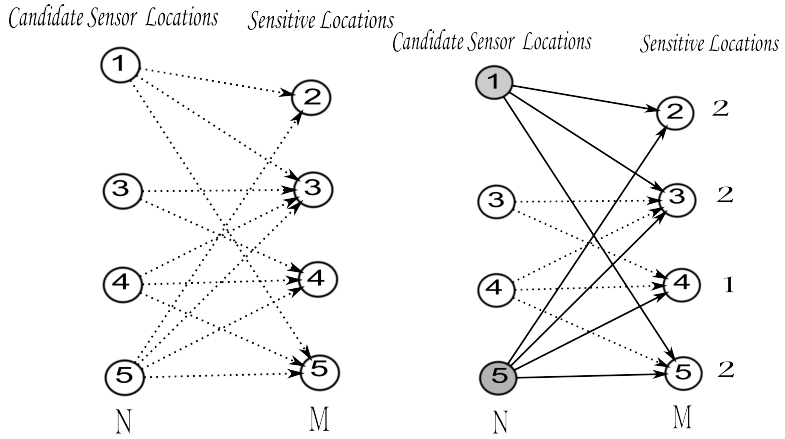
Proof. The proof follows easily by contradiction. Namely, suppose that γ leximin maximal for (5.1) and that $(\phi(\gamma_1), \ldots, \phi(\gamma_m))$ (denoted $\phi(\gamma)$ for short) is not leximin maximal for the corresponding system. Hence, there exists vector $\eta \neq \phi(\gamma)$ that is leximin maximal for the system corresponding to $\phi \circ f$. Therefore, $\phi^{-1}(\eta) \neq \gamma$ is leximin maximal for (5.1), yielding the contradiction. The reverse is shown similarly. □

It is straightforward to see that the above result can be extended to $x \in \mathbb{Z}_+^n$.

5.3 The stochastic equitable sensor location problem

5.3.1 Problem description

The Equitable Sensor Location Problem can be represented by a bipartite graph $G(N, M, A)$ with a set of nodes $N = \{1, \ldots, n\}$ representing candidate sensor locations, a set of nodes $M = \{1, \ldots, m\}$ representing locations that should be protected and a set A of directed links. A link from node $i \in N$ to node $j \in M$ implies that a sensor at i monitors node j. If there is no link from node i to node j, then a sensor at i does not monitor j. We assume that we have K sensors available to be placed in the candidate locations in order to protect the selective locations. Consider the bipartite graph in Figure 5.1a. Suppose



(a) Graph representation of the equitable sensor location problem

(b) Graph representation of the solution obtained by choosing the vertices 1 and 5

Figure 5.1: Graphical representation of the basic problem.

that sensors are located at nodes 1 and 5, as represented in Figure 5.1b. Then, locations $2, 3$ and 5 are monitored by both sensors and location 4 is monitored only by the sensor at node 5. If sensors are located at nodes 1 and 4, locations 3 and 5 are protected by both sensors, and locations 2 and 4 are protected by a single sensor.

We consider in this paper a probabilistic version of the problem, where the effective monitoring of node j by node i is represented by random variable a_{ij}. Specifically, a_{ij} is a Bernoulli random variable that takes value 1 with a given probability p_{ij}. Hence, p_{ij} represents the probability that a sensor at node i detects an intruder at node j under

normal operating conditions. We further assume that the random variables $\{a_{ij}, i \in N, j \in M\}$ are independent. Notice that the case of $p_{ij} = 0$ is represented by the lack of link from i to j in the graph representation. Also, if all detection probabilities are equal to one, the problem reduces to the Equitable Facility Location Problem since each sensitive location is fully protected by a single monitoring sensor.

5.3.2 Mathematical formulation

Let binary optimization variable x_i represent whether or not a sensor is placed in the candidate location i and $q_j(x)$ denote the probability that an intruder is not detected at node j. We obtain

$$\begin{aligned} q_j(x) &= P\left(\sum_{i \in N} a_{ij} x_i < 1\right) \\ &= P\left(a_{ij} x_i < 1, \forall i \in N\right) && (a_{ij} \text{ is binary}) \\ &= \prod_{i \in N} P\left(a_{ij} x_i < 1\right) && (\text{independence}) \\ &= \prod_{i \in N} (1 - p_{ij} x_i) \\ &= \prod_{i \in N} (1 - p_{ij})^{x_i} && (x_i \text{ is binary.}) \end{aligned}$$

We consider two distinct objective functions in this section, the *lexicographic* one and the *proportional* one. Let us first focus on the lexicographic case. With respect to the $q_j(x)$ criterion, system (5.2) can be written as:

$$\left\{ q_j(x) \leq \gamma_j,\, j \in M,\, \sum_{i \in N} x_i = K, x \in \{0,1\}^n \right\}, \tag{5.3}$$

where one looks for a feasible leximax minimal vector γ. The above problem seems hard at first sight since the criteria $q_j(x)$ is clearly non-linear. This is where Proposition 4 comes into play. We can use the logarithmic function as function ϕ, which combined with

the fact that x is a binary solution vector, allows to linearize the functions involved:

$$\log(q_j(x)) = \log\left(\prod_{i\in N}(1-p_{ij})^{x_i}\right) = \sum_{i\in N}(\log(1-p_{ij}))x_i,$$

and system (5.3) becomes

$$\left\{\sum_{i\in N}(\log(1-p_{ij}))x_i \leq \gamma_j,\, j\in M,\, \sum_{i\in N} x_i = K, x\in\{0,1\}^n\right\}.$$

Therefore, computing the leximax minimal vector can be done using the approach shown in [KOW04].

Let us now turn to the problem minimizing the proportional fairness, which is formally defined as

$$\sum_{j\in M}\log(q_j(x)). \tag{5.4}$$

In view of (5.4) above, solving the proportional fair sensor location problem amounts to solve

$$\begin{aligned}
\min\ & \sum_{j\in M}\sum_{i\in N}(\log(1-p_{ij}))x_i \\
\text{s.t.}\ & \sum_{i\in N} x_i = K \\
& x_i \in \{0,1\}, \forall i\in N.
\end{aligned}$$

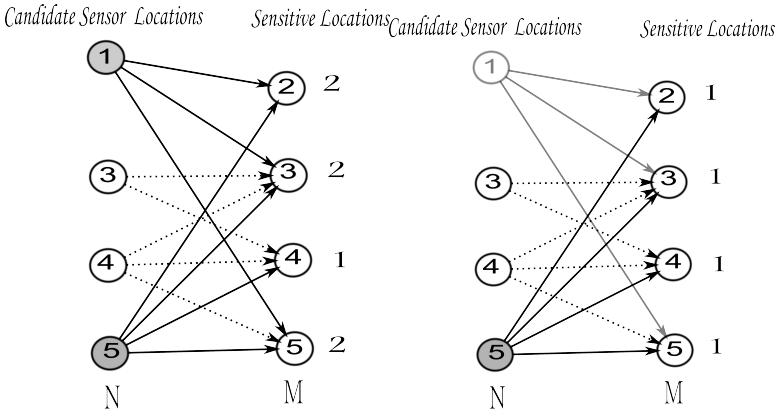
Clearly, the above problem is tractable as it can be solved in $O(|N||M| + |N|\log|N|)$ by ordering the n coefficients $\{\sum_{j\in M}\log(1-p_{ij}), i\in N\}$ in increasing order and choosing the K first elements.

5.3.3 The resilient sensor location problem

This subsection is devoted to the resilient sensor location problem. We consider the problem when some failures can occur and the system needs to be properly dimensioned

in order to cover all possible failure's states.

Figure 1 presents the problem when all four sensors are operating and all links connecting nodes in N to nodes in M are operational. However some of these links may fail. For example all links emanating from node 1 in set N may fail (which is equivalent to a failure of the sensor in node 1). Figure 2 presents the scenario where the sensor at node 1 failed (the dashed links do not provide protection anymore). Under this scenario each of the locations $2, 3, 4$, and 5 is now protected by a single sensor. If a partial failure of the sensor at node 1 occurs, only one or two of the outgoing links from node 1 do not provide protection.



(a) Graphical representation of the solution obtained by choosing the vertices 1 and 5

(b) Graphical representation when all outgoing links from node 1 failed

Figure 5.2: Graphical representation of the resilient sensor location problem.

Let us introduce some additional notation. Let $\mathcal{S} \subseteq 2^K$ be the set of possible failure states, each state S is composed of a subset of sensors in N not operational at the same time; and $q_j^S(x)$ is the value of $q_j(x)$ over state $S \in \mathcal{S}$. In this context, we need to find a solution (a placement of the sensors) such that each location ensures equitable protection level in all the possible states contained in $\mathcal{S}$.

Proceeding as before we obtain the following set:

$$\left\{ q_j^S(x) \leq \gamma_j,\, j \in M,\, S \in \mathcal{S}, \sum_{i \in N} x_i = K, x \in \{0,1\}^n \right\}. \tag{5.5}$$

This system can be handled similarly to system (5.3). The respective proportional fair problem is formulated below:

$$\begin{aligned} \min \sum_{j \in M} \sum_{i \in S; S \in \mathcal{S}} (\log(1 - p_{ij}))x_i \\ \text{s.t.} \sum_{i \in N} x_i = K \\ x \in \{0,1\}^n, \end{aligned}$$

which can be solved in $O(|M||N||S| + |N| \log |N|)$ using a sort algorithm.

5.4 The ambiguous sensor location problem

In this section, we consider an ambiguous variant of the probabilistic sensor location problem, where probabilities p_{ij} are uncertain. This assumption makes sense in practice as the probabilities describe the normal operating conditions of the sensors. These are, however, likely to be affected by many sources of uncertainty, most of which are hard to predict accurately. Consider, for instance the location of surveillance cameras in an airport to secure points of interest. It may happen that some object is placed temporarily between the vision-field of the camera and the point of interest, thus reducing the probability of detecting an intruder in the point of interest.

This is modeled by introducing an ambiguity set that contains the possible probability distributions. Specifically, we are given nominal values and deviations for the probabilities, respectively denoted by $\overline{p}$ and $\hat{p}$, and we assume that p can be any discrete probability measure in the ambiguity set

$$\mathcal{P} := \{p \in \mathbb{R}_+^{n \times m} \mid p_{ij} = \overline{p}_{ij} - \hat{p}_{ij}\xi_i^j, \xi^j \in \Xi^j\},$$

where set $\Xi^j \subset \{0,1\}^n$ is any $0-1$ set. Notice that, since p is a probability measure, we must define $\overline{p}$ and $\hat{p}$ such that $0 \leq p_{ij} \leq 1$ for each $i \in N$,$j \in M$ and $p \in \mathcal{P}$, which

amounts to impose that $0 \leq \hat{p}_{ij} \leq \overline{p}_{ij} \leq 1$ for each $i \in N$ and $j \in M$.

In the ambiguous setting, we replace the probability $q_j(x)$ that an intruder is not detected at node j by the *worst-case* probability that an intruder is not detected at node j, denoted $\mathfrak{q}_j(x)$. Recalling from Section 5.3 that the effective monitoring of node j by node i is represented by the set of independent Bernouilli random variables a_{ij}, we can define $\mathfrak{q}_j(x)$ formally as

$$\mathfrak{q}_j(x) = \max_{p \in \mathcal{P}} P\left(\sum_{i \in N} a_{ij} x_i < 1\right) = \max_{p \in \mathcal{P}} \prod_{i \in N} (1 - p_{ij} x_i).$$

Ambiguity sets have already been used in the context of ambiguous probabilistic constraints [EI06] and distributionally robust optimization [DY10]. The main difference of our approach with these frameworks is that we stick here to ambiguity sets that contain only Bernouilli probability distributions, while the aforementioned works consider sets of continuous distributions that satisfy, for instance, moment-based constraints.

5.4.1 Linearizing the probability

We show in this section how the worst-case probability can be handled by using classical techniques of robust optimization. Given $\xi^j \in \Xi^j_\Gamma$ for each $j \in M$, we denote in the following $q^\xi_j(x)$ as the value of $q_j(x)$ associated to probability distribution given by $p_{ij} = \overline{p}_{ij} - \hat{p}_{ij}\xi^j_i$ for each $i \in N, j \in M$, namely,

$$q^\xi_j(x) = \prod_{i \in N} (1 - (\overline{p}_{ij} - \hat{p}_{ij}\xi^j_i) x_i).$$

Hence by definition, $q^\xi_j(x)$ and $\mathfrak{q}_j(x)$ are linked through

$$\mathfrak{q}_j(x) = \max_{\xi \in \Xi^j} q^\xi_j(x). \tag{5.6}$$

We show below that, using basic properties of logarithmic functions, we can rewrite $q_j^\xi(x)$ as a linear function of ξ:

$$\log(q_j^\xi(x)) = \log\left(\prod_{i\in N}(1-(\overline{p}_{ij}-\hat{p}_{ij}\xi_i^j)x_i)\right) \tag{5.7}$$

$$= \sum_{i\in N}\log(1-(\overline{p}_{ij}-\hat{p}_{ij}\xi_i^j)x_i) \tag{5.8}$$

$$= \sum_{i\in N}\log(1-\overline{p}_{ij}x_i) + \sum_{i\in N}\log\left(1+\frac{\hat{p}_{ij}}{1-\overline{p}_{ij}}\xi_i^j x_i\right) \tag{5.9}$$

$$= \sum_{i\in N}\log(1-\overline{p}_{ij})x_i + \sum_{i\in N}\log\left(1+\frac{\hat{p}_{ij}}{1-\overline{p}_{ij}}\right)\xi_i^j x_i, \tag{5.10}$$

where (5.8) comes from the fact logarithmic function of a product reduces to the product of respective logarithmic functions, (5.9) is obtained when developping the logarithmic of a summation formula and last (5.10) follows from the fact that x_i and ξ_i^j are binary. To simplify notations, we introduce $\overline{\alpha}_{ij} = \log(1-\overline{p}_{ij})$ and $\hat{\alpha}_{ij} = \log(1+\frac{\hat{p}_{ij}}{1-\overline{p}_{ij}})$, yielding

$$\log(q_j^\xi(x)) = \sum_{i\in N}\overline{\alpha}_{ij}x_i + \sum_{i\in N}\hat{\alpha}_{ij}\xi_i^j x_i. \tag{5.11}$$

5.4.2 Proportional fairness

Similarly to (5.4), the proportional fairness considers the logarithm of the worst-case probabilities $\mathfrak{q}_j(x)$. We obtain for each $j \in M$ that

$$\log(\mathfrak{q}_j(x)) = \log\left(\max_{\xi\in\Xi^j} q_j^\xi(x)\right) \tag{5.12}$$

$$= \max_{\xi\in\Xi^j}\log(q_j^\xi(x)) \tag{5.13}$$

$$= \max_{\xi^j\in\Xi^j}\left(\sum_{i\in N}\overline{\alpha}_{ij}x_i + \sum_{i\in N}\hat{\alpha}_{ij}\xi_i^j x_i\right), \tag{5.14}$$

where (5.12) follows from (5.6), (5.13) holds because the logarithm is a monotone increasing function, and (5.14) follows from (5.11). The resulting sensor location problem is a

classical min max robust optimization problem:

$$\min_{x} \left(\sum_{j \in M} \max_{\xi^j \in \Xi^j} \left(\sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{i \in N} \hat{\alpha}_{ij} \xi_i^j x_i \right) \right) \tag{5.15}$$

$$\text{s.t.} \sum_{i \in N} x_i = K \tag{5.16}$$

$$x_i \in \{0,1\}, \forall i \in N. \tag{5.17}$$

where

$$\overline{\alpha}_{ij} = \log(1 - \overline{p}_{ij}) \tag{5.18}$$

$$\hat{\alpha}_{ij} = \log(1 + \frac{\hat{p}_{ij}}{1 - \overline{p}_{ij}}). \tag{5.19}$$

We prove below that the above problem is $\mathcal{NP}$-Hard if the sets Ξ^j are arbitrary. Specifically, assuming that $|M| = 1$ we show that the proportional ambiguous sensor location problem is $\mathcal{NP}$-hard in the weak sense when $|\Xi| = 2$ while the problem is $\mathcal{NP}$-hard in the strong sense when the cardinality of Ξ is part of the input. These results are in line with the complexity results obtained for the robust counterparts of classical polynomially solvable combinatorial optimization problems, see the survey of [ABV10].

To verify the first claim we present in Theorem 4 a polynomial reduction from the *partition problem*, defined as follows. Given a set $L = \{a_1, \ldots, a_{|L|}\}$ of $|L|$ integers, one wants to find a subset S of L of cardinality $|L|/2$ such that $\sum_{l \in S} a_l = \sum_{l \in S \setminus L} a_l$.

Theorem 4. *The partition problem polynomially reduces to the decision version of the proportional ambiguous sensor location problem where* $|\Xi| = 2$.

Consider an instance of the partition problem given by set L, and let us define for each $i \in N$

$$\underline{a}_i = \min \left(a_i, \frac{2}{|L|} \sum_{k \in N} a_k - a_i \right) \text{ and } \overline{a}_i = \max \left(a_i, \frac{2}{|L|} \sum_{k \in N} a_k - a_i \right).$$

We first construct an instance of problem (5.15)–(5.17) that corresponds to the given instance of the partition problem, and show later how to construct the corresponding instance of the proportional ambiguous sensor location problem. Consider $|M| = 1$, $|N| = |L|$, $K = |L|/2$, $\bar{\alpha}_i = \underline{a}_i$ and $\hat{\alpha}_i = \overline{a}_i - \underline{a}_i$ for every $i \in N$. Finally, we define Ξ as $\{\xi^1,\xi^2\}$ where ξ^1 and ξ^2 are defined for each $i \in N$ by

$$\xi_i^1 = \frac{a_i - \underline{a}_i}{\overline{a}_i - \underline{a}_i} \quad \text{and} \quad \xi_i^2 = \frac{\frac{2}{|L|}\sum_k a_k - a_i - \underline{a}_i}{\overline{a}_i - \underline{a}_i}.$$

With these definitions, we see that $\bar{\alpha}_i + \hat{\alpha}_i\xi_i^1 = a_i$ and $\bar{\alpha}_i + \hat{\alpha}_i\xi_i^2 = \frac{2}{|L|}\sum_{k\in N} a_k - a_i$.

Let x be any vector feasible for problem (5.15)–(5.17) and let $S \subset N$ be the set of indices where $x_i = 1$. Then,

$$\max_{\xi\in\Xi} \sum_{i\in N} \overline{\alpha}_{ij}x_i + \sum_{i\in N} \hat{\alpha}_{ij}\xi_i^j x_i = \max\left(\sum_{i\in S} a_i, \sum_{k\in N} a_k - \sum_{i\in S} a_i\right) \tag{5.20}$$

$$= \max\left(\sum_{i\in S} a_i, \sum_{i\in N\setminus S} a_i\right). \tag{5.21}$$

Hence, the instance of the partition problem is a *yes* instance if and only if the optimal solution cost of problem (5.15)–(5.17) is less than or equal to $\frac{\sum_{i\in N} a_i}{2}$.

We construct next an equivalent instance for the proportional ambiguous sensor location problem. First, notice that, due to constraints (5.16), we can add a constant M to all components of $\overline{\alpha}$ without affecting the optimal solution of the problem. Then, using (5.18) and (5.19), we define

$$\overline{p}_i = 1 - e^{\overline{\alpha}_i + M} \tag{5.22}$$

$$\hat{p}_i = (e^{\hat{\alpha}_i} - 1)(1 - \overline{p}_i) \tag{5.23}$$

for each $i \in N$. One readily verifies that choosing

$$M = -\max_{k\in N} \bar{\alpha}_k + \hat{\alpha}_k$$

yields values of $\overline{p}_i$ and $\hat{p}_i$ that satisfy $0 \leq \hat{p}_i \leq \overline{p}_i \leq 1$ for each $i \in N$. Moreover, the input $\overline{p}_i$ and $\hat{p}_i$ can be expressed by a number of digits that is polynomial in the number of digits of the original input.

The second claim is obtained through a reduction from the decision version of the *stable set problem*, presented in Theorem 5. Given a simple graph $G = (V,E)$, where V is the set of vertices and E is the set of edges, a stable set $S \subseteq V$ is a set of vertices such that for all $u,v \in V$ we have that $(uv) \notin E$. Hence, the decision version of the stable set problem can be stated as follows: given a graph G and an integer k, one wants to determine if there is a stable set of cardinality at least k.

Theorem 5. *The decision version of the stable set problem polynomially reduces to the decision version of the proportional ambiguous sensor location problem.*

Corollary 3. *The proportional ambiguous sensor location problem is $\mathcal{NP}$-hard in the weak sense when $|\Xi| = 2$ and in the strong when the cardinality of Ξ is part of the input.*

Consider an instance for the stable set problem, given by the graph $G = (V,E)$ and the integer k. We construct an instance for the proportional ambiguous sensor location problem as follows: $|M| = 1$, $|N| = |V|$, $K = k$, $\bar{p}_i = 1 - e^{\theta}$ and $\hat{p}_i = (e^{\psi} - 1)e^{\theta}$ for each $i \in N$ where θ and ψ are real numbers chosen such that $0 \leq \hat{p}_i \leq \overline{p}_i \leq 1$ for each $i \in N$. Using the previous reformulation and definitions (5.18) and (5.19), we obtain an instance of problem (5.15)–(5.17) defined by $\bar{\alpha}_i = \theta$ and $\hat{\alpha}_i = \psi$ for every $i \in N$. In order to define Ξ, let us consider $ch(e)$ the characteristic vector of an edge $e = (uv) \in E$, meaning $ch(e) \in \{0,1\}^{|V|}$ and $ch_i(e) = 1$ if and only if $i = u$ or $i = v$. Then, we consider the finite uncertainty set $\Xi = \cup_{e \in E} ch(e)$.

Next, we show that there is a stable set of cardinality at least k in the graph G if and only if the optimal solution cost of the proportional ambiguous sensor location problem associated is smaller than or equals to $k\theta + \psi$. Now we look at the two possible cases:

There exists a stable set of cardinality k : Let $S \subseteq V$ be a stable set of cardinality at least k. Consider the solution vector x for the proportional ambiguous sensor

location problem defined as $x_i = 1$ if and only if $i \in S$. Notice that

$$\max_{\xi \in \Xi} \left(\sum_{i \in N} \theta x_i + \sum_{i \in N} \psi \xi_i x_i \right) = \sum_{i \in N} \theta x_i + \max_{e \in E} \sum_{i \in V} \psi c h_i(e) x_i \leq k\theta + \psi$$

There exists no stable set of cardinality k : By contradiction, suppose that

$$\max_{\xi \in \Xi} \left(\sum_{i \in N} \theta x_i^* + \sum_{i \in N} \psi \xi_i x_i^* \right) \leq k\theta + \psi$$

for a solution x^* and we do not have a stable set of cardinality at least k. As x^* is a binary vector we have that

$$\begin{aligned} \max_{\xi \in \Xi} \left(\sum_{i \in N} \theta x_i^* + \sum_{i \in N} \psi \xi_i x_i^* \right) &= \sum_{i \in N} \theta x_i^* + \max_{\xi \in \Xi} \sum_{i \in N} \psi \xi_i x_i^* \\ &= k\theta + \psi \max_{\xi \in \Xi} \sum_{i \in N} \xi_i x_i^* \\ &\leq k\theta + \psi, \end{aligned}$$

which means that

$$\sum_{i \in V} c h_i(e) x_i^* \leq 1$$

for every $e \in E$. Hence $S^* = \{i \in V | x_i^* = 1\}$ is a stable set of G with cardinality at least k.

In view of the above complexity results, we address the problem through mixed-integer linear programming. Hence, assume that each set Ξ^j corresponds to the set of extreme points of a polytope having a compact formulation. Said differently, $\Xi^j = \text{ext}(\{A^j \xi \leq b^j, \xi \geq 0\})$, where the matrix $A^j \in \mathbb{R}^{k \times n}$ and the vector $b^j \in \mathbb{R}^k$ characterize the polytope. Well-known examples of such sets are the budgeted uncertainty set from [BS03]

$$\text{conv}(\Xi_\Gamma^j) := \left\{ 0 \leq \xi_i^j \leq 1, i \in N, j \in M, \sum_{i \in N} \xi_i^j \leq \Gamma \right\},$$

and its extension to multi-band uncertain by [BDR13]. Then, we use classical techniques to reformulate problem (5.15)–(5.17) as a MILP by dualizing the inner maximization problems. Defining $\mathcal{K} = \{1, \ldots, k\}$, we obtain

$$\begin{aligned}
\max_{\xi^j \in \Xi^j} \left(\sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{i \in N} \hat{\alpha}_{ij} \xi_i^j x_i \right) &= \sum_{i \in N} \overline{\alpha}_{ij} x_i + \max_{\xi^j \in \Xi^j} \sum_{i \in N} \hat{\alpha}_{ij} \xi_i^j x_i \\
&= \sum_{i \in N} \overline{\alpha}_{ij} x_i + \max_{\xi^j \in \text{conv}(\Xi^j)} \sum_{i \in N} \hat{\alpha}_{ij} \xi_i^j x_i \\
&= \sum_{i \in N} \overline{\alpha}_{ij} x_i + \left\{ \begin{array}{ll} \min\limits_{u \geq 0} & \sum\limits_{k \in \mathcal{K}} b_k^j u_k \\ \text{s.t.} & \sum\limits_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i, \, i \in N \end{array} \right. \\
&= \left\{ \begin{array}{ll} \min\limits_{u \geq 0} & \sum\limits_{i \in N} \overline{\alpha}_{ij} x_i + \sum\limits_{k \in \mathcal{K}} b_k^j u_k \\ \text{s.t.} & \sum\limits_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i, \, i \in N \end{array} \right. . \qquad (5.24)
\end{aligned}$$

Therefore, the problem minimizing the proportional fairness amounts to solve the following MILP in optimization vectors x and u

$$\begin{aligned}
\min \; & \sum_{j \in M} \sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{j \in M} \sum_{k \in \mathcal{K}} b_k^j u_k \\
\text{s.t.} \; & \sum_{i \in N} x_i = K \\
& \sum_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i, \, i \in N \\
& x_i \in \{0,1\}, \forall i \in N \\
& u \geq 0.
\end{aligned}$$

When A contains non-negative coefficients, an alternative approach proposed by [BS03] and extended in [Pos16] relies on solving a sequence of deterministic problems. Specifically, let us denote by $A^{'}$ the submatrix obtained from A by not considering the upper bounds on ξ and let k' be the number of lines of A' (for instance, $k' = 1$ for the budgeted uncertainty set Ξ_Γ^j). The iterative algorithm proposed in [BS03, Pos16] solves

the above min max robust optimization problem by solving $O((k'm)^{k'm}(nm)^{k'm})$ problems minimizing the proportional fairness with known probabilities. In particular, the min max robust problem is polynomially solvable if k' and m are constant.

5.4.3 Max-min fairness

The ambiguous counterpart of system (5.3) is

$$\left\{ \mathfrak{q}_j(x) \leq \gamma_j,\, j \in M,\, \sum_{i \in N} x_i = K, x \in \{0,1\}^n \right\},$$

where one looks for a feasible leximax minimal vector γ. Notice that, combining (5.14) with (5.24), we obtain immediately the following relation

$$\log(\mathfrak{q}_j(x)) = \left\{ \begin{array}{ll} \min\limits_{u \geq 0} & \sum\limits_{i \in N} \overline{\alpha}_{ij} x_i + \sum\limits_{k \in \mathcal{K}} b_k^j u_k \\ \text{s.t.} & \sum\limits_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i,\, i \in N \end{array} \right. . \tag{5.25}$$

Using again Proposition 4, and replacing $\log(\mathfrak{q}_j(x))$ by the rhs of (5.25),the problem amounts to find the leximax minimal vector γ feasible for system

$$\left\{ \min_{\substack{\sum_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i,\, i \in N \\ u \geq 0}} \sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{k \in \mathcal{K}} b_k^j u_k \leq \gamma_j,\, j \in M,\, \sum_{i \in N} x_i = K, x \in \{0,1\}^n \right\}, \tag{5.26}$$

Then, one readily verifies that, given γ, x is feasible for each constraint

$$\min_{\substack{\sum_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i,\, i \in N \\ u \geq 0}} \left(\sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{k \in \mathcal{K}} b_k^j u_k \right) \leq \gamma_j$$

if and only there exists a vector $u \geq 0$ that satisfies $\sum\limits_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i$ for each $i \in N$ such that x satisfies

$$\sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{k \in \mathcal{K}} b_k^j u_k \leq \gamma_j$$

for each $j \in M$. Hence, we can replace system (5.26) with the following one

$$\left\{ \sum_{i \in N} \overline{\alpha}_{ij} x_i + \sum_{k \in \mathcal{K}} b_k^j u_k \leq \gamma_j, \, j \in M, \right.$$
$$\left. \sum_{i \in N} x_i = K, x \in \{0,1\}^n, \sum_{k \in \mathcal{K}} A_{ki}^j u_k \geq \hat{\alpha}_{ij} x_i, \, i \in N, u \geq 0 \right\}. \tag{5.27}$$

We can finally find a leximax minimal vector γ feasible for system (5.27) using the algorithm proposed in [KOW04].

5.5 Numerical results

In this section, we report on the computation experiments obtained by applying the above models for different variants of the proportional equitable sensor location problems, namely Proportional Fair, Proportional Fair Resilient and Proportional Fair Ambiguous. All experiments have been carried out on an Intel(R) Core(TM) i7 CPU M60, 2.6Hz 4GB Ram machine and all formulations and algorithms were coded in C++, compiled with a GNU G++ 4.5 compiler and IBM CPLEX 12.3. In the rest of the section, we present the benchmark used in our computations as well as different numerical tests with respect to the proportionally fair sensor location problem and the resilient one. We finish with the ambiguous equitable sensor location problem and conclude with some discussion of the obtained results.

5.5.1 Benchmark generation

To the best of our knowledge, there are no probabilistic instances defined for the sensor (facility) location problem, we tested our algorithm on a set of incremental instances generated randomly. We have built 10 instances per scenario (N,M), where we consider 3 different values for N (the set of candidate locations), 10, 20 and 30; and three other values for the sensitive locations to be protected respectively 30, 40 and 50. We consider a quadratic grid of 100×100 as the space where both sensors and points of interest are

placed randomly. We have also considered two possible cost values (30 and 50) which are assigned randomly to each candidate location.

We assume that we have for each instance a number A of high quality sensors that are produced using a new and yet less mature technology. For the instances where $N = 10$ we have 3 of such sensors, for the instances where $N = 20$ we have 4 and in the instances where $N = 30$ we have 5. This will be important for analyzing the resilient solution and will be discussed in more details ahead. Finally, the surveillance probabilities assigned to a candidate sensor for a sensitive location are expressed as the function of its generation and the distance between both locations. Specifically, the probability that a sensor $i \geq A$ protects a location j is valuated as $(1 - \frac{d(i,j)}{\sqrt{2}*100} . \frac{1}{40})$ and for $i < A$ is evaluated as $(1 - \frac{d(i,j)}{\sqrt{2}*100} . \frac{1}{20})$. Hence, the closer a sensor is to a location, the higher is the probability of protection for that location by that sensor. Similarly, the more expensive the sensor is, the better is the protection it provides.

Following above, we have generated 10 instances for each scenario (10,30), (10,40), and (10,50). Next, for scenarios (20,30), (20,40), (20,50) we have taken the first set of instances and added 10 new locations to each of them. The same routine is used to generate (30,30), (30,40), (30,50) using instances for (20,30), (20,40), (20,50).

5.5.2 Proportionally fair sensor location problem

We analyze the quality regarding mean surveillance probability and standard deviation dispersion of the solutions obtained by the model presented for the proportionally fair sensor location problem. Indeed, a proportionally fair solution intends to reach a compromise between two objectives: the max-min fairness which seeks to reduce inequality among the protected locations versus the overall sum of protecting levels.

Table 5.1 presents the results for the proportionally fair sensor location problem. We present the mean and the variance of the surveillance probability according to the number of installed sensors. Notice that, as expected, the surveillance probability is higher as the number of available sensors rises and the obtained solutions have very low variances.

N	M	Protection	Variance
10	30	88.35%	4.69
10	40	94.99%	4.22
10	50	89.17%	5.25
20	30	89.11%	3.70
20	40	95.22%	3.84
20	50	89.95%	3.85
30	30	89.08%	4.84
30	40	95.22%	3.84
30	50	90.17%	3.53

(a) Choosing 5 sensors

N	M	Protection	Variance
10	30	93.14%	3.88
10	40	97.57%	2.27
10	50	94.81%	2.80
20	30	95.10%	2.28
20	40	97.67%	2.41
20	50	95.30%	2.44
30	30	95.17%	2.38
30	40	97.78%	2.16
30	50	95.56%	2.18

(b) choosing 7 sensors

Table 5.1: Results for the proportionally fair sensor location problem. We report the median surveillance value as well as the variance.

Notice that choosing more sensors allows us not only to improve the protection but also to decrease the variance, meaning that discrepancy between the protection levels in the different places are decreasing, turning the solution more fair.

5.5.3 Resilient sensor location problem

Concerning the resilient sensor location problem we tested the same instances presented before under a set of scenarios that represents all the possibilities of failure of 1, 2 or 3 sensors over the A sensors from the new technology. Intuitively, this represents the fact that these sensors provide a larger surveillance probability, but they are unstable. This is the case in many real instances, for example, we can imagine that the three first sensors are "new" or "untested" sensors, they provide better surveillance than the sensors we already have, but they are not fully reliable.

Tables 5.2-5.5, present the results for the resilient sensor location problem. In Table 5.2 we present the mean and the variance of the surveillance probability according to the number of installed sensors for the scenario without any sensor in failure. Notice that the solution obtained by solving the proportinally fair resilient model (called resilient solution) and the solution obtained by solving the proportional fair model (called also

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	88.35%	4.69	80.72%	5.59
10	40	94.99%	4.22	94.70%	3.94
10	50	89.17%	5.25	85.45%	5.68
20	30	89.11%	3.70	87.19%	6.82
20	40	95.22%	3.84	94.93%	4.32
20	50	89.95%	3.85	88.10%	5.00
30	30	89.08%	4.85	88.10%	4.69
30	40	95.22%	3.84	94.55%	5.11
30	50	90.17%	3.53	88.86%	3.98

(a) Choosing 5 sensors

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	93.14%	3.88	90.32%	4.22
10	40	97.57%	2.27	97.05%	2.10
10	50	94.81%	2.80	93.25%	2.66
20	30	95.10%	2.28	93.92%	3.63
20	40	97.67%	2.41	97.64%	2.38
20	50	95.30%	2.44	93.98%	3.48
30	30	95.17%	2.38	94.37%	3.73
30	40	97.78%	2.16	97.63%	2.45
30	50	95.56%	2.18	94.84%	2.50

(b) Choosing 7 sensors

Table 5.2: Results for the resilient sensor location problem. We report the mean surveillance value as well as the variance for the scenario without failures.

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	82.71%	5.38	80.72%	5.59
10	40	93.04%	4.73	94.70%	3.94
10	50	84.16%	5.47	85.45%	5.68
20	30	83.63%	4.84	87.19%	6.82
20	40	95.22%	3.84	94.93%	4.32
20	50	84.83%	4.53	88.10%	5.00
30	30	83.69%	6.13	88.10%	4.69
30	40	95.22%	3.84	94.55%	5.11
30	50	85.01%	4.60	88.86%	3.98

(a) Choosing 5 sensors

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	89.95%	4.66	90.32%	4.22
10	40	96.63%	2.61	97.05%	2.10
10	50	92.46%	2.99	93.25%	2.66
20	30	92.41%	4.03	93.92%	3.63
20	40	96.87%	2.62	97.64%	2.38
20	50	93.04%	2.89	93.98%	3.48
30	40	97.78%	2.16	97.63%	2.45
30	30	92.79%	3.24	94.37%	3.73
30	50	93.32%	2.80	94.84%	2.50

(b) Choosing 7 sensors

Table 5.3: Results for the resilient sensor location problem. We report the mean surveillance value as well as the variance for the worst-case among the three scenarios where one sensor in {1,2,3} fails.

equitable solution) are quite close to each other regarding the surveillance level and even the variation with some advantage to the equitable solution.

Nevertheless, the situation changes when we look at the scenarios where we have some sensors failing. In Table 5.3, we compare between the behavior of the equitable and resilient solutions in scenarios where one "high-tech" sensor fails. Notice that, the surveillance levels are very close, with a little advantage to the resilient solution. In Table 5.4 we have the comparison for the scenario where two "high-tech" sensors fail. We see that the differences between the two models are more perceivable. It is even more so

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	71.33%	6.70	80.72%	5.59
10	40	88.92%	7.27	94.70%	3.94
10	50	84.16%	5.47	85.45%	5.68
20	30	72.64%	7.03	87.19%	6.82
20	40	92.31%	6.17	94.93%	4.32
20	50	84.83%	4.53	88.10%	5.00
30	30	72.96%	8.54	88.10%	4.69
30	40	92.31%	6.17	94.55%	5.11
30	50	85.01%	4.60	88.86%	3.98

(a) Choosing 5 sensors

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	83.58%	6.04	84.14%	5.50
10	40	94.70%	3.94	97.05%	2.10
10	50	89.34%	3.91	93.25%	2.66
20	30	87.19%	6.82	93.92%	3.63
20	40	94.93%	4.32	97.64%	2.38
20	50	93.04%	2.89	93.98%	3.48
30	30	88.10%	4.69	94.37%	3.73
30	40	96.41%	3.57	97.63%	2.45
30	50	93.32%	2.80	94.84%	2.50

(b) Choosing 7 sensors

Table 5.4: Results for the resilient sensor location problem. We report the mean surveillance value as well as the variance for the worst case among the scenarios where two sensors in {1,2,3} fail.

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	57.30%	6.76	80.72%	5.59
10	40	61.82%	6.31	81.98%	5.03
10	50	72.36%	10.96	85.45%	5.68
20	30	72.64%	7.03	87.19%	6.82
20	40	75.44%	5.46	83.86%	4.35
20	50	74.14%	7.36	88.10%	5.00
30	30	72.96%	8.54	88.10%	4.69
30	40	75.44%	5.46	83.68%	5.61
30	50	74.86%	5.87	88.86%	3.98

(a) Choosing 5 sensors

N	M	Prop. Fair		PF Resilient	
		Prot	Var	Prot	Var
10	30	76.09%	5.59	84.14%	5.50
10	40	81.98%	5.03	89.52%	3.94
10	50	81.55%	7.39	88.48%	4.49
20	30	87.19%	6.82	93.92%	3.63
20	40	83.86%	4.35	92.58%	3.11
20	50	88.10%	5.00	93.98%	3.48
30	30	88.10%	4.69	94.37%	3.73
30	40	89.30%	4.06	92.90%	2.95
30	50	88.86%	3.98	94.84%	2.50

(b) Choosing 7 sensors

Table 5.5: Results for the resilient sensor location problem. We report the mean surveillance value as well as the variance for the scenarios where the sensors numbered as 1,2 and 3 fail.

the case where we have three "high-tech" sensors failing represented in Table 5.5.

To conclude, notice that, as expected, the surveillance probability of the resilient solution is higher than the proportional fair solution in the case where we have sensors in failure as well as the variance is smaller. Notice also that, even in the scenario without sensors in failure, and the resilient solution provides a median surveillance probability that is reasonable, while in failure cases it gives solutions notably better comparing to the proportional fair solution.

5.5.4 Ambiguous sensor location problem

Here we adopt the budgeted uncertainty polytope described before. We consider a deviation of 25% for all the probability values on all the instances for all the sensors, except for the sensors numbered from 1 to $|A|$ in which case, the deviation is set to be equal to the probability surveillance, meaning that these sensors can, in a way, become totally inoperational. We set the value 5 for the parameter Γ_s (the maximum number of probabilistic deviations for each sensor s), for all the sensors, except for the first three sensors we have $\Gamma_s = 30$. We computed the Proportional Fair Ambiguous solution for the latter case and compared its behaivor with that of the standard Proportional Fair solution for two different scenarios. The first set of scenarios is built as follows: for each instance we set the probability values at their nominal value (i.e., no deviations are taken into account). For the second scenario we set the surveillance probabilities of each sensor s to their worst value for the lowest Γ_s of them while keep at their nominal value for the rest of probabilities. As we did before, we report the mean surveillance probability values and the standard deviations for both solutions, the basic proportionally fair solution and the proportionally fair ambiguous solution.

N	M	Prop. Fair		PF. Ambiguous	
		Prot	Var	Prot	Var
10	30	88.35%	4.69	80.72%	5.59
10	40	94.99%	4.22	95.45%	3.12
10	50	89.17%	5.25	89.59%	3.81
20	30	89.11%	3.70	87.19%	6.82
30	30	89.08%	4.85	88.10%	4.69
20	40	95.22%	3.84	94.19%	5.21
30	40	95.22%	3.84	93.69%	6.32
20	50	89.95%	3.85	87.73%	5.68
30	50	90.17%	3.53	89.97%	3.88

(a) Choosing 5 sensors

N	M	Prop. Fair		PF. Ambiguous	
		Prot	Var	Prot	Var
10	30	93.14%	3.88	92.06%	3.72
10	40	97.57%	2.27	97.57%	2.27
10	50	94.81%	2.80	94.81%	2.80
20	30	95.10%	2.28	93.92%	3.63
20	40	97.67%	2.41	97.64%	2.38
20	50	95.30%	2.44	94.94%	2.30
30	30	95.17%	2.38	94.37%	3.73
30	40	97.78%	2.16	97.63%	2.45
30	50	95.56%	2.18	95.39%	2.31

(b) Choosing 7 sensors

Table 5.6: Results for the ambiguous sensor location problem, first scenario

Notice that for both scenarios the proportionally fair solution and the ambiguous solution have mean surveillance probabilities close to each other, with a slight advantage

N	M	Prop. Fair		PF. Ambiguous	
		Prot	Var	Prot	Var
10	30	77.78%	26.736	78.44%	5.93
10	40	92.05%	9.293	94.58%	3.56
10	50	86.36%	12.502	88.79%	4.71
20	30	79.59%	27.248	85.36%	7.59
20	40	91.91%	10.651	93.33%	5.76
20	50	87.95%	8.578	87.20%	6.05
30	30	79.78%	27.521	86.51%	5.67
30	40	92.05%	10.652	92.91%	6.62
30	50	87.64%	10.919	89.39%	4.82

(a) Choosing 5 sensors

N	M	Prop. Fair		PF. Ambiguous	
		Prot	Var	Prot	Var
10	30	82.68%	28.28	90.12%	4.09
10	40	94.68%	9.16	97.01%	2.60
10	50	92.48%	8.87	94.02%	3.17
20	30	84.94%	28.90	92.58%	4.11
20	40	95.10%	8.51	97.22%	2.65
20	50	93.32%	8.81	94.62%	2.73
30	30	85.35%	29.04	92.99%	4.65
30	40	95.01%	9.29	97.25%	2.59
30	50	94.00%	6.58	95.05%	2.94

(b) Choosing 7 sensors

Table 5.7: Results for the ambiguous sensor location problem, second scenario

to the ambiguous solution in the second scenario. An interesting result is that the variance is lower in the ambiguous solution for all tested cases of the second scenario, which shows the robustness of such solutions.

Chapter 6

Conclusion

This chapter concludes the thesis and presents a brief summary of the main findings as well as some suggestions for future research on the perspective of what we have done but also in a more general context with respect to Robust Optimisation area. This thesis has had as a primary focus the question of methods for robust optimization and robust lot-sizing as the enlighting application. From our point of view, the main achievements of this thesis stand in exploring the multistage optimization problems under uncertainty, an important class of RO problems, for which some new advances are proposed. In particular, with respect to the budgeted model of Bertsimas and Sim, we have to propose a new dynamic programming algorithm to solve the separation problems which are based on the maximum number of deviations allowed and on the size of such deviations. Another important finding is about the robust multistage optimization problems for which a new relaxation method based on the approach of perfect information is developed.Lot-sizing has been one of the main applications used to illustrate the above developments. We have also tried to make the connection between the fairness and the robustness for a special study case which is the equitable sensor location problem, a special case of equitable facility location problem.

Although, the work presented in this thesis is still an ongoing work and the robust optimization area remains from about 15 years an open research area. With respect to

the work presented in this thesis, several points may have been further explored.

We may cite the need to assess the quality of solutions obtained by the approximation algorithm proposed in Chapter 3 by comparing with an exact method. Another point of interest here is to know if some of the adversarial problems, solved by it, are polynomial. Specifically, the adversarial problem related to the lot-sizing problem. Notice that we know that this problem is polynomial for $\Gamma = |H|$ and Γ fixed, we suspect that this problem is polynomial although we believe that general adversarial problem is $\mathcal{NP}$-hard. Concerning the perfect information relaxation, it will be interesting to find a bound to measure the gap between the exact solution and this obtained with perfect information relaxation. It will also be interesting to study this relaxation for different problems, especially robust integer adjustable problems, where the conventional methods for lower bounding do not work. Concerning the relaxations presented in Chapter 4, we show that must of them are polynomially solvable even for polytopes more complex than the traditionally budgeted uncertainty set. Two questions arise naturally in this context: first, is there a compact linear polytope such that the relaxations presented are $\mathcal{NP}$-hard? Second, is there a property for the uncertainty set that allows solving the adversarial problem polynomially.e, how this egalitarian strategy compares to the simple strategy presented before.

The last point is related to the connection between fairness and robustness. Ensuring fairness intrinsically brings some robustness, but when one includes the uncertainty, this becomes interesting. The work presented in Chapter 5 is restricted to a special application case but it seems easily extendable to the general case.

Bibliography

[ABV10] Hassene Aissi, Cristina Bazgan, and Daniel Vanderpooten. General approximation schemes for min-max (regret) versions of some (pseudo-)polynomial problems. *Discrete Optimization*, 7(3):136–148, 2010.

[ACD13] Agostinho Agra, Marielle Christiansen, and Alexandrino Delgado. Mixed integer formulations for a short sea fuel oil distribution problem. *Transportation Science*, 47(1):108–124, 2013.

[ACF+13] Agostinho Agra, Marielle Christiansen, Rosa M. V. Figueiredo, Lars Magnus Hvattum, Michael Poss, and Cristina Requejo. The robust vehicle routing problem with time windows. *Computers & OR*, 40(3):856–866, 2013.

[AD16] Amir Ardestani-Jaafari and Erick Delage. Robust optimization of sums of piecewise linear functions with application to inventory problems, 2016.

[AK00] P. Alimonti and V. Kann. Some APX-completeness results for cubic graphs. *Theoretical Computer Science*, 237(12):123 – 134, 2000.

[AP14] Bita Analui and Georg Ch. Pflug. On distributionally robust multiperiod stochastic optimization. *Computational Management Science*, 11(3):197–220, 2014.

[AP16] Josette Ayoub and Michael Poss. Decomposition for adjustable robust linear optimization subject to uncertainty polytope. *Computational Management Science*, 13(2):219–239, 2016.

[AR09] A. Agra and C. Requejo. The linking set problem: a polynomial special case of the multiple-choice knapsack problem. *Journal of Mathematical Sciences*, 161(6):919–929, 2009.

[ASNP16] Agostinho Agra, Marcio C. Santos, Dritan Nace, and Michael Poss. A dynamic programming approach for a class of robust optimization problems. *SIAM Journal on Optimization*, 26(3):1799–1823, 2016.

[AYP11] Ayşegül Altın, Hande Yaman, and Mustafa Ç. Pınar. *Performance Models and Risk Management in Communications Systems*, chapter A Hybrid Polyhedral Uncertainty Model for the Robust Network Loading Problem, pages 157–172. Springer New York, New York, NY, 2011.

[BBI14] Frank Baumann, Christoph Buchheim, and Anna Ilyina. Lagrangean decomposition for mean-variance combinatorial optimization. In *Combinatorial Optimization - Third International Symposium, ISCO 2014, Lisbon, Portugal, March 5-7, 2014, Revised Selected Papers*, pages 62–74, 2014.

[BCP14] Alain Billionnet, Marie-Christine Costa, and Pierre-Louis Poirion. 2-stage robust MILP with continuous recourse variables. *Discrete Applied Mathematics*, 170:21–32, 2014.

[BDM+99] Peter Brucker, Andreas Drexl, Rolf H. Möhring, Klaus Neumann, and Erwin Pesch. Resource-constrained project scheduling: Notation, classification, models, and methods. *European Journal of Operational Research*, 112(1):3–41, 1999.

[BDR13] Christina Büsing, Fabio D'Andreagiovanni, and Annie Raymond. Robust optimization under multiband uncertainty. In *12th Cologne-Twente Workshop on Graphs and Combinatorial Optimization, Enschede, Netherlands, May 21-23, 2013.*, pages 35–38, 2013.

[BG12] Dimitris Bertsimas and Vineet Goyal. On the power and limitations of affine policies in two-stage adaptive optimization. *Mathematical programming*, 134(2):491–531, 2012.

[BGGN04] Aharon Ben-Tal, A. Goryashko, E. Guslitzer, and Arkadi Nemirovski. Adjustable robust solutions of uncertain linear programs. *Math. Program.*, 99(2):351–376, 2004.

[BGH92] Dimitri P Bertsekas, Robert G Gallager, and Pierre Humblet. *Data networks*, volume 2. Prentice-Hall International New Jersey, 1992.

[BGNV05] Aharon Ben-Tal, Boaz Golany, Arkadi Nemirovski, and Jean-Philippe Vial. Retailer-supplier flexible commitments contracts: A robust optimization approach. *Manufacturing & Service Operations Management*, 7(3):248–271, 2005.

[BIP10] Dimitris Bertsimas, Dan Andrei Iancu, and Pablo A. Parrilo. Optimality of affine policies in multistage robust optimization. *Math. Oper. Res.*, 35(2):363–394, 2010.

[BL97] John R. Birge and François Louveaux. *Introduction to stochastic programming*. Springer series in operations research. Springer, New York, 1997.

[BL11] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. 2011.

[BMdO15] Sergio VB Bruno, Leonardo AM Moraes, and Welington de Oliveira. Optimization techniques for the brazilian natural gas network planning problem. *Energy Systems*, pages 1–21, 2015.

[BN98] Aharon Ben-Tal and Arkadi Nemirovski. Robust convex optimization. *Math. Oper. Res.*, 23(4):769–805, 1998.

[BÖ08] Daniel Bienstock and Nuri Özbay. Computing robust basestock levels. *Discrete Optimization*, 5(2):389–414, 2008.

[BS03] Dimitris Bertsimas and Melvyn Sim. Robust discrete optimization and network flows. *Math. Program.*, 98(1-3):49–71, 2003.

[BS04] Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[BS07] Hans-Georg Beyer and Bernhard Sendhoff. Robust optimization – a comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196(33–34):3190 – 3218, 2007.

[BT06] Dimitris Bertsimas and Aurélie Thiele. A robust optimization approach to inventory theory. *Operations Research*, 54(1):150–168, 2006.

[BTEGN09] A. Ben-Tal, L. El Ghaoui, and A.S. Nemirovski. *Robust Optimization*. Princeton Series in Applied Mathematics. Princeton University Press, October 2009.

[CH92] Gregory F. Cooper and Edward Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[CPL13] CPLEX. *IBM ILOG CPLEX 12.5 Reference Manual*, 2013.

[CT08] Ann Melissa Campbell and Barrett W. Thomas. Probabilistic traveling salesman problem with deadlines. *Transportation Science*, 42(1):1–21, 2008.

[CTC07] Mei-Shiang Chang, Ya-Ling Tseng, and Jing-Wen Chen. A scenario planning approach for the flood emergency logistics preparation problem under uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 43(6):737–754, 2007.

[DY10] Erick Delage and Yinyu Ye. Distributionally robust optimization under moment uncertainty with application to data-driven problems. *Operations Research*, 58(3):595–612, 2010.

[EI06] Emre Erdogan and Garud Iyengar. Ambiguous chance constrained problems and robust optimization. *Math. Program.*, 107(1-2):37–61, 2006.

[GCCP93] BG Gorenstin, NM Campodonico, JP Costa, and MVF Pereira. Power system expansion planning under uncertainty. *IEEE Transactions on Power Systems*, 8(1):129–136, 1993.

[GdH13] Bram L. Gorissen and Dick den Hertog. Robust counterparts of inequalities containing sums of maxima of linear functions. *European Journal of Operational Research*, 227(1):30–43, 2013.

[GH86] Erol Gelenbe and Georges Hébrail. A probability model of uncertainty in data bases. In *Proceedings of the Second International Conference on Data Engineering, February 5-7, 1986, Los Angeles, California, USA*, pages 328–333, 1986.

[GJ98] Yu Gang and Yang Jian. On the robust shortest path problem. *Computers & OR*, 25(6):457–468, 1998.

[GKHR03] Nicole Growe-Kuska, Holger Heitsch, and Werner Romisch. Scenario reduction and scenario tree construction for power management problems. In *Power Tech Conference Proceedings, 2003 IEEE Bologna*, volume 3, pages 7–pp. IEEE, 2003.

[GLS93] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1993.

[GM08] Yongpei Guan and Andrew Miller. *A Polynomial Time Algorithm for the*

Stochastic Uncapacitated Lot-Sizing Problem with Backlogging, pages 450–462. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[GMT14] Virginie Gabrel, Cécile Murat, and Aurélie Thiele. Recent advances in robust optimization: An overview. *European Journal of Operational Research*, 235(3):471 – 483, 2014.

[God00] Michel Godet. The art of scenarios and strategic planning: tools and pitfalls. *Technological forecasting and social change*, 65(1):3–22, 2000.

[Gus02] Elana Guslitser. *Uncertainty-immunized solutions in linear programming*. PhD thesis, TECHNION-ISRAEL INSTITUTE OF TECHNOLOGY, 2002.

[GdH15] Bram L. Gorissen, İhsan Yanıkoğlu, and Dick den Hertog. A practical guide to robust optimization. *Omega*, 53:124 – 137, 2015.

[HA09] Kai Huang and Shabbir Ahmed. The value of multistage stochastic programming in capacity planning under uncertainty. *Operations Research*, 57(4):893–904, 2009.

[HCF10] Yongxi Huang, Chien-Wei Chen, and Yueyue Fan. Multistage optimization of the supply chains of biofuels. *Transportation Research Part E: Logistics and Transportation Review*, 46(6):820–830, 2010.

[HLW01] Kjetil K Haugen, Arne Løkketangen, and David L Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operational Research*, 132(1):116–122, 2001.

[IK75] Oscar H. Ibarra and Chul E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *J. ACM*, 22(4):463–468, 1975.

[IK88] Toshihide Ibaraki and Naoki Katoh. *Resource Allocation Problems: Algorithmic Approaches*. MIT Press, Cambridge, MA, USA, 1988.

[ISS13] Dan Andrei Iancu, Mayank Sharma, and Maxim Sviridenko. Supermodularity and affine policies in dynamic robust optimization. *Operations Research*, 61(4):941–956, 2013.

[JPTL13] Bin Jiang, Jian Pei, Yufei Tao, and Xuemin Lin. Clustering uncertain data based on probability distribution similarity. *IEEE Trans. Knowl. Data Eng.*, 25(4):751–763, 2013.

[KN08] Olivier Klopfenstein and Dritan Nace. A robust approach to the chance-constrained knapsack problem. *Oper. Res. Lett.*, 36(5):628–632, 2008.

[KOW04] Michael M Kostreva, Włodzimierz Ogryczak, and Adam Wierzbicki. Equitable aggregations and multiple criteria analysis. *European Journal of Operational Research*, 158(2):362–377, 2004.

[KSLS13] Diego Klabjan, David Simchi-Levi, and Miao Song. Robust stochastic lot-sizing by means of histograms. *Production and Operations Management*, 22(3):691–710, 2013.

[KWG11] Daniel Kuhn, Wolfram Wiesemann, and Angelos Georghiou. Primal and dual linear decision rules in stochastic and robust optimization. *Mathematical Programming*, 130:177–209, 2011.

[LB05] Jean-Yves Le Boudec. Rate adaptation, congestion control and fairness: A tutorial. *Web page, November*, 2005.

[LF14] Zukui Li and Christodoulos A. Floudas. Optimal scenario reduction framework based on distance of uncertainty distribution and output performance: I. single reduction via mixed integer linear optimization. *Computers & Chemical Engineering*, 70:50 – 66, 2014.

[LNPS16] Hanan Luss, Dritan Nace, Michael Poss, and Marcio Costa Santos. Equitable sensor location problems. *ROADEF*, 2016.

[Lus12] Hanan Luss. *Equitable Resource Allocation: Models, Algorithms and Applications*, volume 101. John Wiley & Sons, 2012.

[LZX+15] Jinfei Liu, Haoyu Zhang, Li Xiong, Haoran Li, and Jun Luo. Finding probabilistic k-skyline sets on uncertain data. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 1511–1520, New York, NY, USA, 2015. ACM.

[Min09] Michel Minoux. On robust maximum flow with polyhedral uncertainty sets. *Optimization Letters*, 3(3):367–376, 2009.

[MPS13] Michele Monaci, Ulrich Pferschy, and Paolo Serafini. Exact solution of the robust knapsack problem. *Computers & OR*, 40(11):2625–2631, 2013.

[MS04] John M. Mulvey and Bala Shetty. Financial planning via multi-stage stochastic optimization. *Computers & OR*, 31(1):1–20, 2004.

[NLK08] Arnie Neidhardt, Hanan Luss, and KR Krishnan. Data fusion and optimal placement of fixed and mobile sensors. In *Sensors Applications Symposium, 2008. SAS 2008. IEEE*, pages 128–133. IET, 2008.

[NO07] Dritan Nace and James B Orlin. Lexicographically minimum and maximum load linear programming problems. *Operations research*, 55(1):182–187, 2007.

[NP08] Dritan Nace and Michal Pióro. Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial. *IEEE Communications Surveys & Tutorials*, 10(4):5–17, 2008.

[Ogr97] Włodzimierz Ogryczak. On the lexicographic minimax approach to location problems. *European Journal of Operational Research*, 100(3):566–585, 1997.

[OLP+14] Wlodzimierz Ogryczak, Hanan Luss, Michał Pióro, Dritan Nace, and Artur Tomaszewski. Fair optimization and networks: A survey. *Journal of Applied Mathematics*, 2014, 2014.

[Pos14] Michael Poss. Robust combinatorial optimization with variable cost uncertainty. *European Journal of Operational Research*, 237(3):836–845, 2014.

[Pos16] Michael Poss. Easy uncertainty sets for robust combinatorial optimization. 2016. Working paper.

[PP15] Georg Ch. Pflug and Alois Pichler. Dynamic generation of scenario trees. *Computational Optimization and Applications*, 62(3):641–668, 2015.

[PW94] Yves Pochet and Laurence A Wolsey. Polyhedra for lot-sizing with wagner—whitin costs. *Mathematical Programming*, 67(1-3):297–323, 1994.

[PW06] Yves Pochet and Laurence A Wolsey. *Production planning by mixed integer programming*. Springer Science & Business Media, 2006.

[RM10] Deepti Rani and Maria Madalena Moreira. Simulation–optimization modeling: a survey and potential application in reservoir systems operation. *Water resources management*, 24(6):1107–1138, 2010.

[Roc01] R.T. Rockafella. Optimazation over uncertainty. *Lecture Notes - University of Washigton*, 2001.

[SCL12] Oguz Solyali, Jean-François Cordeau, and Gilbert Laporte. Robust inventory routing under demand uncertainty. *Transportation Science*, 46(3):327–340, 2012.

[Sha11] Alexander Shapiro. A dynamic programming approach to adjustable robust optimization. *Oper. Res. Lett.*, 39(2):83–87, 2011.

[Tem13] Horst Tempelmeier. *Stochastic Lot Sizing Problems*. Springer New York, 2013.

[TG15] Z. Melis Teksan and Joseph Geunes. A polynomial time algorithm for convex cost lot-sizing problems. *Oper. Res. Lett.*, 43(4):359–364, 2015.

[Tim15] Anna V. Timonina. Multi-stage stochastic optimization: the distance between stochastic scenario processes. *Computational Management Science*, 12(1):171–195, 2015.

[TK04] S Armagan Tarim and Brian G Kingsman. The stochastic dynamic production/inventory lot-sizing problem with service-level constraints. *International Journal of Production Economics*, 88(1):105–119, 2004.

[Tsa03] Edward P. K. Tsang. *Constraint Based Scheduling: Applying Constraint Programming to Scheduling Problems*, volume 6. 2003.

[VJM16] Goran Vojvodic, Ahmad I. Jarrah, and David P. Morton. Forward thresholds for operation of pumped-storage stations in the real-time energy market. *European Journal of Operational Research*, 254(1):253 – 268, 2016.

[WLO+11] Haixun Wang, Shijun Li, Satoshi Oyama, Xiaohua Hu, and Tieyun Qian, editors. *Web-Age Information Management - 12th International Conference, WAIM 2011, Wuhan, China, September 14-16, 2011. Proceedings*, volume 6897 of *Lecture Notes in Computer Science*. Springer, 2011.

[Yag88] Ronald R Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on systems, Man, and Cybernetics*, 18(1):183–190, 1988.

[Zhu10] Yuanguo Zhu. Uncertain optimal control with application to a portfolio selection model. *Cybernetics and Systems*, 41(7):535–547, 2010.

[ZZ11] Bo Zeng and L Zhao. Solving two-stage robust optimization problems by a constraint-and-column generation method. *University of South Florida, FL, Tech. Rep*, 2011.