# Contributions to robust combinatorial optimization with budgeted uncertainty

Michael Poss

Université de Montpellier

Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier

Thèse d'habilitation à diriger des recherches

présentée par

Michael Poss

spécialité : informatique

# Contributions to robust combinatorial optimization with budgeted uncertainty

| | | |
|---|---|---|
| Christian Artigues | Directeur de recherche, LAAS | (Rapporteur) |
| Cristina Bazgan | Professeur, Université Paris Dauphine | (Examinateur) |
| Walid Ben-Ameur | Professeur, Telecom SudParis | (Examinateur) |
| Rodolphe Giroudeau | Maître de conférences, Université de Montpellier | (Examinateur) |
| Arie Koster | Professeur, RWTH Aachen University | (Rapporteur) |
| Andrea Lodi | Professeur, Polytechnique Montréal | (Rapporteur) |

Soutenue le 22 novembre 2016



# Acknowledgements

Je souhaite tout d'abord remecrier Christian Artigues, Arie Koster et Andrea Lodi d'avoir accepté d'être rapporteurs pour mon habilitation à diriger des recherches. Merci aussi à Cristina Bazgan, Walid Ben-Ameur et Rodolphe Giroudeau d'avoir accepté de participer au jury.

Cette habilitation n'aurait pas de raison d'être sans les collaborations que j'ai eues ces dernières années. Parmi les excellents chercheurs avec qui j'ai eu le privilège de collaborer, je me permets de distinguer Agostinho Agra, Marin Bougeret, Dritan Nace et Artur Alves Pessoa. J'ai beaucoup appris auprès d'eux, tout en appréciant leurs immenses qualités humaines. Artur et Marin m'ont initié aux algorithmes d'approximation qui deviennent lentement mais surement un nouveau centre d'intérêt. Agostinho et Dritan m'ont accompagné dans ma première expérience d'encadrement de thèse, celle de Marcio Costa Santos, que je remercie également pour avoir été un excellent étudiant, motivé et d'une bonne humeur contagieuse.

Je remercie également Zacharie Alès, Quentin Botton, Annie Chateau, Sébastien Destercke, Boris Detienne, Luigi Di Puglia Pugliese, Bernard Fortz, Luis Gouveia, Francesca Guerriero, Didier Josselin, Martine Labbé, François Louveaux, Enrico Natalizio, Michal Pióro, Sara Mattia, Christian Raack, Cristina Requejo, Claudia Sagastizábal, Luidi Simonetti, Eduardo Uchoa, François Vanderbeck et Mathias Weller pour les fructueuses collaborations que nous avons menées, menons et mènerons ensemble.

Un grand merci aux collègues des différents laboratoires par lesquels je suis passé. Je remercie en particulier Antoine Jouglet et Jacques Carlier de chez Heudiasyc, qui ont contribué à rendre mon séjour picard très agréable, ainsi que les membres d'ESPACE, du LIA et du LIRMM qui m'ont accueilli à bras ouverts.

Enfin et surtout, merci à Rosa, ainsi qu'à Carolina et Felipe, pour avoir rendu ma vie si riche.

# Foreword

The objective of this manuscript is to provide a coherent survey of the work I have done on robust combinatorial optimization since I defended my PhD thesis in February 2011. Hence, the manuscript does not intend to cover all work realized since my PhD defense and in particular, it does not depict my work on affine routing and robust network design, which is detailed in Ayoub and Poss [2016], Pessoa and Poss [2015], Pioro et al. [2016], Poss and Raack [2013].

I started working on robust combinatorial optimization during my postdoctoral stay at the University of Aveiro, in Portugal. There I met Agostinho Agra and Cristina Requejo, who kindly invited me to the project on robust maritime transportation they were working on at that time. The results of the collaboration are partly described in Chapter 6. In parallel to this work, several discussions with A. Agra, C. Requejo as well as Lars Magnus Hvattum and Rosa Figueiredo led me to introduce the variable budgeted uncertainty set described in Chapter 2. I continued working on robust combinatorial optimization when I joined Heudiasyc in October 2012. More specifically, I started advising Marcio Costa Santos with Agostinho Agra and Dritan Nace on robust lot-sizing problems. Some of the thesis results are presented in Chapters 5 and 6. At that time I had the chance to meet Luigi di Puglia Pugliese and Francesca Guerriero with whom I have been working on the robust shortest path problem described in Chapter 3.

I joined team MAORE in Feburary 2015, whose reseach topics include approximation algorithms for scheduling problems. It has therefore been natural to combine my recent experience on robust optimization with their knowledge of scheduling to propose new robust scheduling models. Marin Bougeret has joined me in this research, and together with Artur Alves Pessoa, we have provided some initial results on robust scheduling, some of which are summarized in Chapter 4.

# Contents

**Acknowledgements** **i**

**Foreword** **iii**

**1 Introduction** **3**
1.1 Robust combinatorial optimization . . . . . 4
1.2 Budgeted uncertainty . . . . . 7
1.3 Literature review . . . . . 9
1.4 Structure and contributions of the thesis . . . . . 10

**2 Variable uncertainty** **13**
2.1 Introduction . . . . . 13
2.2 Definition . . . . . 13
2.3 Dualization . . . . . 16
2.4 Iterative algorithm . . . . . 17
2.5 Cost uncertainty . . . . . 17
2.6 Numerical experiments . . . . . 21

**3 Constrained shortest path** **25**
3.1 Introduction . . . . . 25
3.2 $\mathcal{NP}$-hardness for general uncertainty sets . . . . . 26
3.3 Complexity of $\mathcal{U}^{\Gamma}$-$CSP$ and $\mathcal{U}^{\gamma}$-$CSP$ . . . . . 27
3.4 Complexity of $\mathcal{U}^{\Gamma}$-$TWSP$ and $\mathcal{U}^{\gamma}$-$TWSP$ . . . . . 28
3.5 Label-setting algorithm . . . . . 31
3.6 Computational experiments . . . . . 34

**4 Scheduling** **37**
4.1 Introduction . . . . . 37
4.2 Minimizing sum of completion times . . . . . 38
4.3 Minimizing makespan on identical machines . . . . . 44
4.4 Minimizing makespan on unrelated machines . . . . . 46

**5 Lot-sizing** **51**
5.1 Introduction . . . . . 51
5.2 Decomposition algorithm . . . . . 52
5.3 Dynamic programming algorithm . . . . . 53
5.4 Fully polynomial time approximation scheme . . . . . 56
5.5 Computational experiments . . . . . 59

**6 Vehicle routing with time windows** **63**
6.1 Introduction . . . . . 63
6.2 Left and right hard . . . . . 66
6.3 Left and right soft . . . . . 71
6.4 Left soft and right hard . . . . . 73
6.5 Left hard and right soft . . . . . 74

**7 Conclusion and future work** **79**
7.1 Numerical research directions . . . . . 79
7.2 Theoretical research directions . . . . . 80
7.3 Lot-sizing . . . . . 81
7.4 Beyond budgeted uncertainty . . . . . 82

**Bibliography** **82**

# Chapter 1

# Introduction

Robust optimization is an increasingly popular approach to handle the uncertainty that arises in mixed-integer linear optimization problems. Unlike stochastic programming, which requires exact knowledge of the probability distributions and leads to very large-scale optimization problems, robust optimization only describes the variability of the uncertain parameters through bounded uncertainty sets. When the uncertainty set can be described by polynomial numbers of linear inequalities and constraints, the robust counterparts can be addressed efficiently through MILP formulations whose dimensions are comparable to the dimensions of the deterministic problem. In a sense, robust optimization overcomes the two main drawbacks of stochastic programming, which are the necessity of knowing exactly the behavior of the uncertain parameters and the dimensions of the resulting reformulations.

Although the dimensions of the robust reformulations are controlled, they tend the break the combinatorial structure of the problems at hand. For instance, the robust reformulations of problems that have totally unimodular formulations are usually not totally unimodular anymore. Another drawback of the classical techniques is that they are essentially limited to problems whose constraints are defined by linear functions of the uncertain parameters. This prevents the approach from being applied to lot-sizing problems, scheduling and routing problems with soft time windows, among others.

A fundamental breakthrough in robust combinatorial optimization came with the budgeted uncertainty set from Bertsimas and Sim [2003]. Budgeted uncertainty leads to robust problems that can be solved by solving a polynomial numbers of deterministic problems whenever the robust constraints (and/or the objective) are defined by linear functions of the uncertain parameters. In particular, budgeted uncertainty keeps the complexity of a large class of combinatorial optimization problems. In addition to its good computational properties, the budgeted uncertainty set benefits from profound links with individual probabilistic constraints.

Our objectives in this thesis are to build around the results from Bertsimas and Sim [2003], along the three following directions. First, we extend the budgeted uncertainty set

to a more general uncertainty paradigm that keeps the property of budgeted uncertainty, namely the controlled theoretical complexity of the robust counterparts and the link with probabilistic constraints. Second, we study the limit of the positive results from Bertsimas and Sim [2003] from a theoretical viewpoint, identifying problems for which the robust counterparts are more difficult than the deterministic versions. Third, we propose decomposition algorithms that split a robust combinatorial optimization problems into a relaxed master problem and separation problems. The master problem essentially handles the combinatorial structure of the problem, while the separation problems take care of the robust constraints. These algorithms can handle non-linearities present in lot-sizing problems, and scheduling and routing problems with soft time windows.

Sections 1.1 and 1.2 state precisely the technical context of the thesis. Specifically, Section 1.1 defines formally robust combinatorial optimization, recalls the limitations of the general model and introduce classical results in the field. Section 1.2 then introduces the budgeted uncertainty set from Bertsimas and Sim [2003], states the main complexity results and briefly introduces the link with probabilistic constraints. Section 1.3 provides a short survey on robust optimization, focusing on the works addressing robust combinatorial optimization with budgeted uncertainty. The contributions and the structure of the thesis are further described in Section 1.4.

## 1.1 Robust combinatorial optimization

Discrete optimization problems can often be formulated as follows. Let $\mathcal{X} \subseteq \{0,1\}^n$ be a given feasibility set and $c$ be a given cost vector in $\mathbb{R}^n$. Then, the discrete/combinatorial optimization problem described by $(\mathcal{X}, c)$ is

$$CO \qquad \min_{x \in \mathcal{X}} c^T x. \tag{1.1}$$

In this thesis, we are interested by problem $CO$ under uncertainty, which means that the cost and/or the feasibility set of $CO$ are uncertain. Formally, $c$ and the parameters that describe $\mathcal{X}$ are not known with precision, but instead, belong to known uncertainty sets, falling into the framework of Robust Optimization [Ben-Tal et al., 2009]. For the sake of uniformity, we assume hereafter that all uncertain parameters are components of the vector/matrix $u$, whose dimensions depend on the context. In particular, the above uncertain cost vector $c$ is renamed $u_0$ in this chapter. We further decompose $CO$ by splitting the structure of $\mathcal{X}$ into components $\mathcal{X}^{comb}$ and $\mathcal{X}^{num}$ where $\mathcal{X}^{comb} \subseteq \{0,1\}^n$ contains the combinatorial structure of $\mathcal{X}$ while $\mathcal{X}^{num}$ is defined by linear constraints

$$u_j^T x \leq b_j, \quad j = 1, \ldots, m, \tag{1.2}$$

where $u$ is a $(m+1) \times n$ matrix and $b$ a $m$-dimensional vector. Hence, we redefine $\mathcal{X}$ as $\mathcal{X} = \mathcal{X}^{comb} \cap \mathcal{X}^{num}$. Some of the results and problems studied in the thesis focus on cost

uncertainty, which is represented by setting $m = 0$. In that case, $\mathcal{X}^{comb} = \mathcal{X}$ and $\mathcal{X}^{comb}$ contains all constraints of the problem, and $u$ becomes a $n$-dimensional vector rather than a matrix.

The uncertainty we are studying in this thesis is known as *numerical uncertainty* [Ben-Tal et al., 2009], which means that only the numerical constraints (1.2) and the cost vector $u_0$ are affected by uncertainty. In contrast, the combinatorial structure of the problem, $\mathcal{X}^{comb}$, is known with precision. Given *finite* uncertainty sets $\mathcal{U}_j$ for each $j = 0, \ldots, m$, the robust counterpart of problem $CO$ is

$$\mathcal{U}\text{-}CO \qquad \min \Big\{ \quad \max_{u_0 \in \mathcal{U}_0} u_0^T x \ : \tag{1.3}$$

$$u_j^T x \leq b_j, \quad j = 1, \ldots, m, \ u_j \in \mathcal{U}_j, \tag{1.4}$$

$$x \in \mathcal{X}^{comb} \quad \Big\}.$$

Problem $\mathcal{U}$-$CO$ is affected by uncertainty in two ways. First, its objective function (1.3) optimizes the worst-case cost function $\max_{u_0 \in \mathcal{U}_0} u_0^T x$. Second, each binary vector $x \in \mathcal{X}^{comb}$ must satisfy constraints (1.4) for all values of $(u_1, \ldots, u_m) \in \mathcal{U}_1 \times \cdots \times \mathcal{U}_m$.

From the theoretical viewpoint, $\mathcal{U}$-$CO$ is in general harder than $CO$. For instance, if $\mathcal{X}^{comb}$ represents all paths from a given source to a given destination and $m = 0$, we obtain the shortest path problem, well-known to be solvable in polynomial time when the costs are positive. However, its robust version defined for an arbitrary set $\mathcal{U}_0$ is $\mathcal{NP}$-hard [Yu and Yang, 1998], even in the strong sense if the cardinality of $\mathcal{U}_0$ is unbounded. Similar results arise for other easy instances of problems $CO$, such as the assignment problem and the spanning tree problem; see Aissi et al. [2009] and the references therein.

In spite of its theoretical difficulty, problem $CO$ can be solved fairly efficiently whenever each uncertainty set can be described as the set of extreme points of a polytope with a compact description. The first way to do this relies on the fundamental theorem of robust optimization due to Ben-Tal and Nemirovski [1998] and recalled below. For simplicity, the theorem is stated for $m = 0$, that is, assuming that the problem does not contain robust constraints; one readily extends it to general problems $\mathcal{U}$-$CO$.

**Theorem 1.** *Let $m = 0$ and consider $\alpha \in \mathbb{R}^{l \times n}$ and $\beta \in \mathbb{R}^l$ that define polytope*

$$\mathcal{P} := \{u_0 \in \mathbb{R}^n_+ \ : \ \alpha_k^T u_0 \leq \beta_k, \ k = 1, \ldots, l\},$$

*such that $\mathcal{U}_0$ corresponds to the set of extreme points of $\mathcal{P}$. Problem $\mathcal{U}$-$CO$ is equivalent to the following mixed-integer linear program, with an additional vector of optimization*

*variables denoted by z*

$$\min\Big\{ \quad \sum_{k=1}^{l} \beta_k z_k \ : \\ \sum_{k=1}^{l} \alpha_{ki} z_k \geq x_i, \quad i = 1, \ldots, n, \\ z \geq 0, x \in \mathcal{X}^{comb} \quad \Big\}.$$

*Proof.* The proof follows directly by the dualizing the inner maximization of $\mathcal{U}$-$CO$. Namely

$$\min_{x \in \mathcal{X}^{comb}} \max_{u_0 \in \mathcal{U}_0} u_0^T x = \min_{x \in \mathcal{X}^{comb}} \min \left\{ \sum_{k=1}^{l} \beta_k z_k \ : \sum_{k=1}^{l} \alpha_{ki} z_k \geq x_i, i = 1, \ldots, n, z \geq 0 \right\},$$

and the result is obtained by regrouping the two min. □

A natural algorithmic approach to $\mathcal{U}$-$CO$ consists in solving the reformulation stated in Theorem 1. A different approach to problem $\mathcal{U}$-$CO$ is based on a cutting-plane algorithm that generate elements of $\mathcal{U}_0, \ldots, \mathcal{U}_m$ on the fly. Specifically, given subsets $\mathcal{U}_j^* \subset \mathcal{U}_j$ for each $j = 0, \ldots, m$, the algorithm relies on the following relaxed master problem

$$\mathcal{U}^*\text{-}CO \qquad \min \left\{ \max_{u_0 \in \mathcal{U}_0^*} u_0^T x \ : u_j^T x \leq b_j, \ j = 1, \ldots, m, u_j \in \mathcal{U}_j^*, x \in \mathcal{X}^{comb} \right\}.$$

Given a solution $x^*$ to the master problem, and its cost denoted by $z^*$, the algorithm alternates between solving $\mathcal{U}^*$-$CO$ and solving the separation problems

$$\max_{u_0 \in \mathcal{U}_0} u_0^T x^* \text{ and } \max_{u_j \in \mathcal{U}_j} u_j^T x^* \text{ for each } j = 1, \ldots, m. \tag{1.5}$$

If $\max_{u_0 \in \mathcal{U}_0} u_0^T x^* > z^*$ or there are indices $j$ for which $\max_{u_j \in \mathcal{U}_j} u_j^T x^* > b_j$, then the corresponding vectors $u_0$ and $u_j$ are added to $\mathcal{U}_0^*$ and $\mathcal{U}_j^*$, respectively, and the master problem is solved again. Otherwise, the algorithm returns the optimal solution $x^*$. When each set $\mathcal{U}_j$ corresponds to the set of extreme points of a polytope with a compact description, (1.5) amount to solve $m + 1$ linear programs. The aforementioned iterative algorithm can of course be improved in various ways, the first of which replaces the iterative procedure by a branch-and-cut algorithm where the separation problems are called only at specific nodes in the branch-and-bound tree, as commonly done in Benders decomposition algorithms (e.g. Botton et al. [2013]). The relative performance of the cutting-plane algorithm and Theorem 1 depends on the context, see the comparisons realized in Bertsimas et al. [2015], Fischetti and Monaci [2012], Monaci et al. [2013], Pessoa and Poss [2015], among others.

From the numerical viewpoint, Theorem 1 and the cutting-plane algorithm have provided efficient ways to solve $\mathcal{U}$-$CO$ assuming that each uncertainty set corresponds to the

set of extreme points of a polytope with a compact description. Yet, these approaches usually break the combinatorial structure of the problem at hand. For instance, applying Theorem 1 to the shortest path problem yields a mixed-integer linear program that does not have the integrality property. This, and the fact that arbitrary uncertainty sets make these problems difficult from the theoretical viewpoint, have motivated the introduction of the structured uncertainty sets described in the next section.

## 1.2 Budgeted uncertainty

Given two vectors $\overline{v} \in \mathbb{R}^n$ and $\hat{v} \in \mathbb{R}^n$, and a positive integer $\Gamma > 0$, Bertsimas and Sim [2003, 2004] introduce the budgeted uncertainty set as

$$\mathcal{U}^\Gamma := \left\{ v \in \mathbb{R}^n : v_i = \overline{v}_i + \xi_i \hat{v}_i, \xi \in \{0,1\}^n, \sum_{i=1}^{n} \xi_i \leq \Gamma \right\},$$

which means that, in any scenario $u \in \mathcal{U}^\Gamma$, at most $\Gamma$ components of $u$ will take their extreme values. The counterpart of $\mathcal{U}$-$CO$ with budgeted uncertainty is

$$\mathcal{U}^\Gamma\text{-}CO \qquad \min \left\{ \max_{u_0 \in \mathcal{U}_0^\Gamma} u_0^T x \ : u_j^T x \leq b_j,\ j = 1, \ldots, m, u_j \in \mathcal{U}_j^\Gamma, x \in \mathcal{X}^{comb} \right\},$$

where $\mathcal{U}_0^\Gamma, \ldots, \mathcal{U}_m^\Gamma$ are obtained from $\mathcal{U}^\Gamma$ by replacing $\overline{v}$ and $\hat{v}$ by the vectors $\overline{u}_j \in \mathbb{R}^n$ and $\hat{u}_j \in \mathbb{R}^n$ for each $j = 0, \ldots, m$, respectively, and the value of $\Gamma$ is assumed to be identical for all sets $\mathcal{U}_0^\Gamma, \ldots, \mathcal{U}_m^\Gamma$ to simplify notations.

Set $\mathcal{U}^\Gamma$ is motivated by two fundamental results proved in Bertsimas and Sim [2003] and Bertsimas and Sim [2004], respectively. First, the set yields a problem $\mathcal{U}^\Gamma$-$CO$ that belongs to the complexity class of its counterpart $CO$ whenever the number of uncertain constraints $m$ is constant. This property is formally stated in the following theorem, proved by Bertsimas and Sim [2003] for $m = 0$ and independently extended to positive values of $m$ by Álvarez-Miranda et al. [2013] and Goetzmann et al. [2011].

**Theorem 2.** *The optimal solution to the robust problem $\mathcal{U}^\Gamma$-$CO$ can be obtained by taking the minimum among the optimal solution costs of $(n+1)^{m+1}$ deterministic problems, defined as*

$$\hat{u}_{0\ell_0}\Gamma + \min \left\{ \sum_{i=1}^{n} \breve{u}_{0i} x_i \ : \sum_{i=1}^{n} \breve{u}_{ji}^{\ell_j} x_i \leq \breve{b}_j^{\ell_j},\ j = 1, \ldots, m, x \in \mathcal{X}^{comb} \right\}.$$

*for each $(\ell_0, \ldots, \ell_m) \in \{1, \ldots, n+1\}^{m+1}$, where $\breve{b}_j^{\ell_j} = b_j - \Gamma \hat{u}_{j\ell_j}$ and $\breve{u}_{ji}^{\ell_j} = \overline{u}_{ji} + \max(\hat{u}_{ji} - \hat{u}_{j\ell_j}, 0)$ for each $i = 1, \ldots, n$, $j = 0, \ldots, n$ and $\ell_j = 1, \ldots, n+1$, and $\hat{u}_{j\ell_{n+1}} = 0$.*

*Proof.* We provide a short proof for the case $m = 0$. Problem $\mathcal{U}^\Gamma$-$CO$ can be rewritten as follows

$$\min_{x\in\mathcal{X}^{comb}} \max_{u_0\in\mathcal{U}_0^\Gamma} u_0^T x = \min_{x\in\mathcal{X}^{comb}} \left( \overline{u}_0^T x + \max_{\substack{\xi\in[0,1]^n \\ \sum_i \xi_i\le\Gamma}} \sum_{i=1}^n \xi_i \hat{u}_{0i} x_i \right) \tag{1.6}$$

$$= \min_{x\in\mathcal{X}^{comb}} \left( \overline{u}_0^T x + \min_{\substack{\theta+y_i\ge\hat{u}_{0i}x_i \\ \theta,y\ge 0}} \Gamma\theta + \sum_{i=1}^n y_i \right) \tag{1.7}$$

$$= \min_{x\in\mathcal{X}^{comb}} \left( \overline{u}_0^T x + \min_{\theta\ge 0} \Gamma\theta + \sum_{i=1}^n \max(\hat{u}_{0i}x_i - \theta, 0) \right) \tag{1.8}$$

$$= \min_{x\in\mathcal{X}^{comb}} \left( \overline{u}_0^T x + \min_{\theta\ge 0} \Gamma\theta + \sum_{i=1}^n x_i \max(\hat{u}_{0i} - \theta, 0) \right) \tag{1.9}$$

where (1.6) holds because polytope $\{\xi \in [0,1]^n, \sum_i \xi_i \le \Gamma\}$ has the integrality property, (1.7) follows from linear programming duality, (1.8) is obtained by substituting $y_i$ with $\max(\hat{u}_{0i}x_i - \theta, 0)$, and (1.9) holds because $x$ is binary. Notice then that the minimum of $\theta$ in (1.9) is always reached at one element of $\{\hat{u}_{01}, \ldots, , \hat{u}_{0n}, 0\}$ (see Bertsimas and Sim [2003] for a detailed explanation). Therefore, (1.9) is equal to

$$\min_{\theta\in\{\hat{u}_{01},\ldots,\hat{u}_{0n},0\}} \min_{x\in\mathcal{X}^{comb}} \left( \Gamma\theta + \sum_{i=1}^n x_i(\overline{u}_{0i} + \max(\hat{u}_{0i} - \theta, 0)) \right),$$

proving the result. □

Whenever $m$ is a constant that does not depend on the instance (e.g. for the knapsack problem, $m = 1$), Theorem 2 implies that $\mathcal{U}^\Gamma$-$CO$ can be solved by solving a polynomial number of problems $CO$. The following result is the pendant of Theorem 2 for the approximation ratio achievable for problem $\mathcal{U}^\Gamma$-$CO$. Again, it was first proved by Bertsimas and Sim [2003] in the case $m = 0$ and extended to constant values of $m$ by Goetzmann et al. [2011].

**Theorem 3.** *Assume that $m$ is a constant. If $CO$ has an $\rho$-approximation polynomial-time algorithm, then $\mathcal{U}^\Gamma$-$CO$ is also $\rho$-approximable.*

Theorems 2 and 3 radically changed the landscape of robust combinatorial optimization since many instances of polynomially solvable problems $CO$ also had polynomially solvable robust counterparts (e.g. spanning tree problems, shortest path problems, minimum cut problems, and assignment problems) contrasting with the $\mathcal{NP}$-hardness of general robust counterparts [Aissi et al., 2009]. In fact, the generality of Theorem 2 has raised the following question: *is there an instance of problem $CO$ that is (pseudo-) polynomially solvable whose counterpart $\mathcal{U}^\Gamma$-$CO$ is (strongly) $\mathcal{NP}$-hard?* We shall see in the balance

of this manuscript that the answer to the question is *yes*, more specifically: the shortest path problem with time windows, pseudo-polynomially solvable, and minimizing the weighted sum of completion times, polynomially solvable, both turn strongly $\mathcal{NP}$-hard in the robust context.

The second fundamental property behind $\mathcal{U}^\Gamma$ concerns its relation with probabilistic constraints. Let $\tilde{u}_i = \overline{u}_i + \eta_i \hat{u}_i$ be the random variable associated with parameter $u_i$ and suppose that $\eta_i$, $i \in \{1, \ldots, n\}$, are arbitrary random variables independently and symmetrically distributed in $[-1, 1]$. Bertsimas and Sim [2004] construct different functions $\beta : \mathbb{R}^2 \to \mathbb{R}$ such that any vector $x$ that satisfies the robust constraint $\sum_{i=1}^n u_i x_i \leq b$, for all $u \in \mathcal{U}^\Gamma$, also satisfies

$$P\left(\sum_{i=1}^{n} \tilde{u}_i x_i > b\right) \leq \beta(\Gamma, n). \tag{1.10}$$

Hence, given a probability level $\epsilon \in (0, 1)$, choosing $\Gamma$ such that $\beta(\Gamma, n) \leq \epsilon$ ensures that the probabilistic constraint $P\left(\sum_{i=1}^n \tilde{u}_i x_i > b\right) \leq \epsilon$ will be satisfied. This is a very strong result since robust constraints are significantly easier to handle than individual probabilistic constraints. Bertsimas and Sim [2004] present various examples of bounds $\beta(\Gamma, n)$. The strongest of these bounds is given by

$$\beta(\Gamma, n) = \frac{1}{2^n}\left((1-\mu)\binom{n}{\lfloor\nu\rfloor} + \sum_{l=\lfloor\nu\rfloor+1}^{n}\binom{n}{l}\right) \tag{1.11}$$

where $\nu = (\Gamma + n)/2$ and $\mu = \nu - \lfloor\nu\rfloor$. Bertsimas and Sim [2004] also present a weaker bound

$$\beta^{weak}(\Gamma, n) = \exp\left(-\frac{\Gamma^2}{2n}\right), \tag{1.12}$$

that yields an analytical definition for $\Gamma$ as a function of $\epsilon$ and $n$, namely $\Gamma^{weak}(\epsilon, n) = (-2n\ln(\epsilon))^{\frac{1}{2}}$.

## 1.3 Literature review

We review next some of the main works on robust optimization, focusing on those that address robust combinatorial optimization with budgeted uncertainty. This section is by no means an exhaustive survey on robust combinatorial optimization with general uncertainty sets, for which we refer to Aissi et al. [2009] and Kouvelis and Yu [2013], among others. It is even less a review on the robust optimization literature, for which we refer to Ben-Tal et al. [2009], Bertsimas et al. [2011], Gabrel et al. [2014] and the references therein.

Robust optimization is usually traced back to the work of Soyster [1973] who introduces a robust model where parameters belong to bounded and convex sets. His model can be

reformulated as a linear program by taking the maximal values of the parameters over the uncertainty sets.

An important step forward has independently been taken by Ben-Tal and Nemirovski [1998] and El Ghaoui et al. [1998] who study linear programs with ellipsoidal uncertainty sets. They show that their models can be reformulated as conic quadratic programs. Although their models are more complex than the one of Soyster, they are less conservative and lead to cheaper robust solutions. More recently, Bertsimas and Sim [2004] have introduced the aforementioned uncertainty sets $\mathcal{U}^\Gamma$ whose conservatism is comparable to the conservatism obtained with the ellipsoids from Ben-Tal and Nemirovski [1998], El Ghaoui et al. [1998].

Differently from these works, which study mathematical programs under uncertainty, Kouvelis and Yu [2013] study $CO$ under discrete uncertainty sets and address the problems from a combinatorial perspective. Their combinatorial approach has been used in subsequent works by various authors; see the survey from Aissi et al. [2009]. In spite of their interest, $\mathcal{U}$-$CO$ problems defined for uncertainty sets of unbounded cardinality are usually more difficult than their deterministic counterparts, and more often than not, the robust counterparts of polynomial $CO$ problems are $\mathcal{NP}$-hard.

In this context, the results of Álvarez-Miranda et al. [2013], Bertsimas and Sim [2003], Goetzmann et al. [2011] opened a new avenue of research, by showing how uncertainty set $\mathcal{U}^\Gamma$ keeps the complexity and approximability properties of the deterministic counterparts. Complementing these general results, researchers have studied particular instances of problem $\mathcal{U}^\Gamma$-$CO$. Klopfenstein and Nace [2008] have studied dynamic programming algorithms for the robust knapsack problem, see also Monaci et al. [2013] for efficient implementations. Combinatorial approaches different from dynamic programming have also been applied to $\mathcal{U}^\Gamma$-$CO$. Büsing et al. [2011] study the combinatorial properties of the recoverable robust knapsack problem to provide a pseudo-polynomial mixed-integer linear programming reformulation. Monaci and Pferschy [2013] study the worst-case optimal solution cost of the robust knapsack problem and propose a greedy heuristic whose performance is also studied from a worst-case point of view. The nice computational properties of $\mathcal{U}^\Gamma$ have led Büsing et al. [2013] to consider multiple deviations for each individual parameter, which they call multiband uncertainty. Based on this, Claßen et al. [2015] propose dynamic programming algorithms for the robust knapsack problem with multiband uncertainty.

## 1.4 Structure and contributions of the thesis

We detail in what follows the contributions of each chapter. While Chapter 2 describes a new uncertainty paradigm, applicable to any problem $CO$, the subsequent four chapters presents complexity results and solution algorithms devised for specific instances of $CO$.

Chapter 2 presents an extension of the budgeted uncertainty set $\mathcal{U}^\Gamma$ that is based on

a function $\gamma$ of the optimization variables, rather than a constant $\Gamma$. We explain in the chapter how this model is motivated by the aforementioned probabilistic result. Further, whenever $\gamma$ is an affine function, the new model keeps the main tractability properties of $\mathcal{U}^\Gamma$ such as the dualization from Theorem 1 and the iterative algorithm presented in Theorem 2. Numerical experiments are presented, confirming the interest of the model from a practical viewpoint.

The subsequent four chapters present solution algorithms for specific robust optimization problems. The algorithms presented in these chapters are of three kinds: exact combinatorial algorithms, approximation algorithms, and Benders-like decomposition algorithms based on mixed-integer linear programming formulations. In a way, the decomposition algorithms try to extend the simplistic cutting plane algorithm presented in Section 1.1 to robust optimization problems that are more complex than $\mathcal{U}^\Gamma$-$CO$ due to the presence of non-linear robust constraints. Hence, as in Section 1.1, these algorithms keep the combinatorial structure $\mathcal{X}^{comb}$ in the relaxed master problem, while the separation problems handle the robust constraints. These chapters present exact dynamic programming algorithms for the separation problems described therein. Notice that the combinatorial algorithms proposed in Chapters 3 and 4 can also handle variable uncertainty, albeit at a higher computational cost. This is not the case for the decomposition algorithms presented in Chapters 5 and 6 as explained therein.

Let us detail more specifically the content of each of these four chapters. Chapter 3 focuses on the resource-constrained shortest path problem, considering either capacity constraints or time windows. We show that the former is $\mathcal{NP}$-hard in the weak sense while the later is $\mathcal{NP}$-hard in the strong sense, contrasting with the deterministic context for which both problems are $\mathcal{NP}$-hard in the weak sense. We propose then an extension of the label-setting algorithm to the robust context, introducing new robust labels. While having an exponential running-time in general, our algorithms are pseudo-polynomial when $\Gamma$ is constant.

Chapter 4 turns to scheduling problems. We consider problems on a single machine that minimize the (weighted and unweighted) sum of completion times and problems that minimize the makespan on parallel and unrelated machines. We provide polynomial algorithms and approximation algorithms: constant factor and average non-constant factor. In addition, we prove that the robust version of minimizing the weighted completion time on a single machine is $\mathcal{NP}$-hard in the strong sense.

Chapter 5 studies lot-sizing problems which, strictly speaking, are not combinatorial optimization problems because the decision variables are real. However, we show in the chapter that the problem can be efficiently addressed through Benders-like decomposition algorithms, where the separation problem becomes a combinatorial optimization problem that can be solved in pseudo-polynomial time through dynamic programming. We then show how reducing the precision on the components of $\hat{u}$ yields a fully polynomial-time approximation scheme. Numerical experiments highlight the supremacy of the dynamic

programming algorithm over naive mixed-integer linear formulations for the separation problem.

Chapter 6 addresses vehicle routing problems with hard and/or soft time windows. We investigate the four problems obtained by combining soft/hard left time windows with soft/hard right time windows. For all problems, we propose Benders-like decomposition algorithms that come down to row-and-column generation algorithms. The separation algorithms are studied for the four problems, turning out to be polynomial in the case of two hard time windows and pseudo-polynomial in the other cases. Numerical experiments are realized for the first case, showing that, with these algorithms, the robust models are essentially as easy to solve as their deterministic counterparts.

The thesis is finally concluded in Chapter 7, where some ongoing and future research directions are discussed. For brevity reasons, some proofs are omitted from the manuscript. We would like to point out that the particular $\mathcal{U}^\Gamma$-robust optimization problems studied in Chapters 3–6 were studied for the first time by the author of the thesis. For that reason, and to keep this manuscript as concise as possible, these chapters only contain few related papers; the interested reader is referred to the papers cited at the beginning of each chapter for comprehensive literature surveys.

# Chapter 2

# Variable uncertainty

*The chapter is based on Poss [2013, 2014].*

## 2.1 Introduction

As mentioned in Chapter 1, one of the main motivations behind set $\mathcal{U}^\Gamma$ is given by the probabilistic bound

$$P\left(\sum_{i=1}^{n} \tilde{u}_i x_i > b\right) \leq \beta(\Gamma, n),$$

which shows how a robust constraint can be used to approximate an ambiguous probabilistic constraint. We show in this chapter how the approximation can be improved by considering a more general paradigm of uncertainty. Our main idea is to let $\Gamma$ depend on the cardinality of feasible solutions, by introducing an uncertainty multifunction $\mathcal{U}^\gamma$. We define formally $\mathcal{U}^\gamma$ in the next section. Section 2.3 shows how the resulting model can be solved by dualizing the robust constraints. Section 2.4 extends the iterative algorithm from Theorem 2 to special cases of $\mathcal{U}^\gamma$. Section 2.5 is devoted to cost uncertainty, defined by setting $m$ to 0. The new model is illustrated numerically in Section 2.6.

## 2.2 Definition

Consider a robust constraint of the form

$$\sum_{i=1}^{n} u_i x_i \leq b, \quad u \in \mathcal{U}^\Gamma. \tag{2.1}$$

We will need in this chapter to consider $\mathcal{U}^\Gamma$ as a polytope rather than a finite set, so that we commit an abuse of language and redefine $\mathcal{U}^\Gamma$ as follows

$$\mathcal{U}^\Gamma := \left\{ u \in \mathbb{R}^n : u_i = \overline{u}_i + \xi_i \hat{u}_i, \xi \in [0,1]^n, \sum_{i=1}^{n} \xi_i \leq \Gamma \right\}.$$

Because robust constraint (2.1) is concave in $u$ (even linear), the maximum over $\mathcal{U}^\Gamma$ is always achieved at one of the extreme points of $\mathcal{U}^\Gamma$ so that the binding constraints in (2.1) always correspond to extreme points of $\mathcal{U}^\Gamma$. Hence, considering $\mathcal{U}^\Gamma$ as a polytope or as the set of its extreme points does not affect the feasibility region characterized by constraint (2.1).

Polytope $\mathcal{U}^\Gamma$ might sometimes be over-conservative because its definition is independent from the value of $x$. Because of this, binary vectors with few non-zero components are more protected than binary vectors with larger numbers of non-zero components. For instance, consider two binary vectors $x^1$ and $x^2$ feasible for constraint (2.1) and suppose that $\|x^1\|_1 := \sum_{i=1}^n x_i^1 = \Gamma$ while $\|x^2\|_1 = 2\Gamma$. The robust constraints associated to $x^1$ and $x^2$ are equivalently written

$$\sum_{i=1}^n u_i x_i^1 = \sum_{i:x_i^1=1} u_i \leq b, \quad u \in \mathcal{U}^\Gamma, \tag{2.2}$$

and

$$\sum_{i=1}^n u_i x_i^2 = \sum_{i:x_i^2=1} u_i \leq b, \quad u \in \mathcal{U}^\Gamma, \tag{2.3}$$

respectively. In a relative point of view, vector $x^1$ is more protected than vector $x^2$ since it is ensured that constraint (2.2) is feasible against the simultaneous deviation of all of its terms while constraint (2.3) is only protected against the simultaneous deviation of half of its terms. This relative point of view has a natural probabilistic interpretation. If $\tilde{u}_i$ are random variables arbitrarily distributed between $\overline{u}_i - \hat{u}_i$ and $\overline{u}_i + \hat{u}_i$, then the probability that constraint $\sum_{i:x_i^1=1} \tilde{u}_i \leq b$ be violated is equal to zero, regardless from the specific distributions of $\tilde{u}$, while the probability that $\sum_{i:x_i^2=1} \tilde{u}_i \leq b$ be violated can be strictly positive for particular choices of $b$ and probability distributions.

To avoid this conservatism, we introduce below a novel model of uncertainty. Instead of considering an uncertainty set $\mathcal{U}^\Gamma \subseteq \mathbb{R}^n$ independent of $x$, we introduce a multifunction of $x$ (point-to-set mapping) $\mathcal{U}^\gamma : \{0,1\}^n \rightrightarrows \mathbb{R}^n$ that is a generalization of the budgeted uncertainty. Given a non-negative function $\gamma : \{0,1\}^n \to \mathbb{R}_+$, we define the variable budgeted uncertainty as

$$\mathcal{U}^\gamma(x) := \left\{ u \in \mathbb{R}^n \ : \ u_i = \overline{u}_i + \xi_i \hat{u}_i, \xi \in [0,1]^n, \sum_{i=1}^n \xi_i \leq \gamma(x) \right\}. \tag{2.4}$$

If $\gamma$ is constantly equal to $\Gamma$, then $\mathcal{U}^\gamma(x)$ coincide with $\mathcal{U}^\Gamma$ for any $x$. In general however, $\mathcal{U}^\gamma$ avoids overprotecting vectors with few components, yielding a less conservative model than $\mathcal{U}^\Gamma$. The pendant of (2.1) for the variable budgeted uncertainty is

$$\sum_{i=1}^n u_i x_i \leq b, \quad u \in \mathcal{U}^\gamma(x). \tag{2.5}$$

We explain next how function $\gamma$ can be defined according to the probabilistic bounds recalled in Chapter 1. It is not hard to see from the proof of Theorem 2 from Bertsimas and Sim [2004] that, for any $x \in \{0,1\}^n$,

$$P\left(\sum_{i=1}^{n} \tilde{u}_i x_i > b\right) \leq \beta(\Gamma, n). \tag{2.6}$$

can be changed to

$$P\left(\sum_{i=1}^{n} \tilde{u}_i x_i > b\right) \leq \beta(\Gamma, \|x\|_1). \tag{2.7}$$

Inequality (2.7) tells us that the value of $\Gamma$ can be changed according to the cardinality of $x$ while ensuring the same level of probability protection. Namely, we see that the smallest value of $\Gamma$ such that the resource constraint is satisfied with probability $1-\epsilon$ is given by the solution of the minimization problem

$$\min\{\Gamma \ : \ \beta(\Gamma, \|x\|_1) \leq \epsilon\}, \tag{2.8}$$

which can be solved by a bisection method. Hence, a natural choice for the function $\gamma$ used in $\mathcal{U}^\gamma$ is given by the function $\gamma^\beta : \{0,1\}^n \to \mathbb{R}_+$, defined as

$$\gamma^\beta(x) = \min\{\Gamma \ : \ \beta(\Gamma, \|x\|_1) \leq \epsilon\}. \tag{2.9}$$

The interest of $\mathcal{U}^{\gamma^\beta}$ is explained below. The minimum value of $\Gamma$ dictated by (2.6) is given by the optimal solution cost of

$$\gamma^\beta(\mathbf{1}) = \min\{\Gamma \ : \ \beta(\Gamma, n) \leq \epsilon\},$$

where $\mathbf{1} = (1, \ldots, 1) \in \mathbb{R}^n$. By definition of $\Gamma$, any vector $x$ that satisfies the robust constraint $\sum_{i=1}^n u_i x_i \leq b$, for all $u \in \mathcal{U}^\Gamma$, also satisfies

$$P\left(\sum_{i=1}^{n} \tilde{u}_i x_i > b\right) \leq \epsilon. \tag{2.10}$$

Now, by construction of $\gamma^\beta$, any vector $x$ that satisfies the robust constraint

$$\sum_{i=1}^{n} u_i x_i \leq b, \quad u \in \mathcal{U}^{\gamma^\beta}, \tag{2.11}$$

also satisfies (2.10). Since $\mathcal{U}^{\gamma^\beta}(x) \subseteq \mathcal{U}^\Gamma$ for all $x \in \{0,1\}^n$, robust constraint (2.11) is less conservative than the classical one given by (2.1).

## 2.3 Dualization

It is not difficult to extend Theorem 1 to point-to-set mappings, but the resulting mixed-integer programs contain non-linear constraints. To avoid this, we approximate $\gamma^\beta$ with a set of affine functions as follows. Let $\gamma^1, \ldots, \gamma^m$ be affine functions of $x$ defined by $\gamma^j(x) = \gamma_0^j + \sum_{i=1}^n \gamma_i^j x_i$. For each $x \in \{0,1\}^n$, the set $\mathcal{U}^{\gamma^1 \cdots \gamma^m}(x)$ contains vectors $u \in \mathbb{R}^n$ such that $u_i = \overline{u}_i + \xi_i \hat{u}_i$ and

$$\begin{aligned} 0 \;&\leq\; \xi_i \;\leq\; 1, && i = 1, \ldots, n, && (2.12)\\ \sum_{i=1}^n \xi_i \;&\leq\; \gamma^j(x), && j = 1, \ldots, m. && (2.13) \end{aligned}$$

In what follows, we will use multifunction $\mathcal{U}^{\gamma^1 \cdots \gamma^m}$ as an approximation of $\mathcal{U}^{\gamma^\beta}$. To ensure that $\mathcal{U}^{\gamma^1 \cdots \gamma^m}$ yields the same probabilistic guarantee as $\mathcal{U}^{\gamma^\beta}$, functions $\gamma^j$, $j = 1, \ldots, m$, must be greater than or equal to $\gamma^\beta$ for all $x \in \{0,1\}^n$ so that $\mathcal{U}^{\gamma^\beta}(x) \subseteq \mathcal{U}^{\gamma^1 \cdots \gamma^m}(x)$ for all $x$. This is formalized in the following lemma, provided without proof.

**Lemma 1.** *Let $\gamma^1, \ldots, \gamma^m$ be affine functions of $x$ such that $\gamma^j(x) \geq \gamma^\beta(x)$ for all $x \in \{0,1\}^n$. If $x^* \in \{0,1\}^n$ satisfies the robust constraint $\sum a_i x_i^* \leq b$, for all $a \in \mathcal{U}^{\gamma^1 \cdots \gamma^m}(x)$, then $P(\sum \tilde{u}_i x_i^* > b) \leq \epsilon$.*

The next result shows how to handle the upper approximation provided by $\gamma^1, \ldots, \gamma^m$.

**Theorem 4.** *Consider robust constraint*

$$\begin{aligned} u^T x \;&\leq\; b, && u \in \mathcal{U}^{\gamma^1 \cdots \gamma^m}(x),\\ x \;&\in\; \{0,1\}^n, \end{aligned} \qquad (2.14)$$

*and suppose that $\gamma^1, \ldots, \gamma^m$ are affine functions of $x$, non-negative for all $x \in \{0,1\}^n$. Then, (2.14) is equivalent to*

$$\sum_{i=1}^n \overline{u}_i x_i + \sum_{j=1}^m \left( \gamma_0^j z_j + \sum_{i=1}^n \gamma_i^j w_{ji} \right) + \sum_{i=1}^n p_i \leq b \qquad (2.15)$$

$$\sum_{j=1}^m z_j + p_i \geq \hat{u}_i x_i, \qquad i = 1, \ldots, n, \qquad (2.16)$$

$$w_{ji} - z_j \geq -\max_j(\hat{u}_j)(1 - x_i), \quad i = 1, \ldots, n,$$

$$j = 1, \ldots, m, \qquad (2.17)$$

$$p, w, z \geq 0, \qquad (2.18)$$

$$x \in \{0,1\}^n. \qquad (2.19)$$

The proof of Theorem 4, omitted for brevity, follows closely the line of the proof of Theorem 1. The main difference is that, after dualizing the adversary problem, one ends

up with products between $\gamma^j(x)$ and the dual variables. Since $\gamma^j$ are affine functions of $x$, the products can be linearized through big-$M$ coefficients, which happen to be equal to $\max_j(\hat{u}_j)$ in our context.

## 2.4 Iterative algorithm

Let us extend the definition of $\mathcal{U}^\Gamma$-$CO$ to $\mathcal{U}^\gamma$:

$$\mathcal{U}^\gamma\text{-}CO \qquad \min\left\{\max_{u_0\in\mathcal{U}_0^\gamma} u_0^T x \ : u_j^T x \le b_j,\ j=1,\dots,m, u_j\in\mathcal{U}_j^\gamma, x\in\mathcal{X}^{comb}\right\}.$$

We see next that the computational complexity of $\mathcal{U}^\gamma$-$CO$ for an *affine function* $\gamma$ is essentially the same as the complexity of $\mathcal{U}^\Gamma$-$CO$.

**Theorem 5.** *Let $\gamma(x)=\gamma_0+\sum_{i=1}^n\gamma_i x_i$ be affine. The optimal solution to the robust problem $\mathcal{U}^\gamma$-$CO$ can be obtained by taking the minimum among the optimal solution cost of $(n+1)^{m+1}$ deterministic problems, defined as*

$$\hat{u}_{0\ell_0}\gamma^0+\min\left\{\sum_{i=1}^n \breve{u}_{0i}x_i \ : \sum_{i=1}^n \breve{u}_{ji}^{\ell_j}x_i\le \breve{b}_j^{\ell_j},\ j=1,\dots,m, x\in\mathcal{X}^{comb}\right\}.$$

*for each $(\ell_0,\dots,\ell_m)\in\{1,\dots,n+1\}^{m+1}$, where $\breve{b}_j^{\ell_j}=b_j-\Gamma^0\hat{u}_{j\ell_j}$ and $\breve{u}_{ji}^{\ell_j}=\overline{u}_j+\gamma_j\hat{u}_j+\max(\hat{u}_{ji}-\hat{u}_{j\ell_j},0)$ for each $i=1,\dots,n$, $j=0,\dots,n$ and $\ell_j=1,\dots,n+1$, and $\hat{u}_{j\ell_{n+1}}=0$.*

The only difference between the problems solved in Theorem 5 and those solved in Theorem 2 lies in the costs coefficients of $x$. In Theorem 5, these coefficients depend on the components $\gamma_i$ of the budget function $\gamma$ while they are independent of $\Gamma$ in Theorem 2.

## 2.5 Cost uncertainty

We assume in this section that $m=0$ and focus on combinatorial optimization problem of the form

$$CO^{cost} \qquad \min_{x\in\mathcal{X}^{comb}} u^T x.$$

We focus here on functions $\gamma$ such that $\gamma(x^1)=\gamma(x^2)$ for each pair of binary vectors that satisfy $\|x^1\|_1=\|x^2\|_1$. Therefore, we commit a slight abuse of notations and assume in this section that the domain of the function $\gamma$ is $\{0,\dots,n\}$ and we redefine $\mathcal{U}^\gamma$ as

$$\mathcal{U}^\gamma(x):=\left\{u\in\mathbb{R}^n \ : \ u_i=\overline{u}_i+\xi_i\hat{u}_i, \xi\in[0,1]^n, \sum_{i=1}^n\xi_i\le\gamma(\|x\|_1)\right\}.$$

Similarly, $\gamma^\beta$ is redefined as $\gamma^\beta:\{0,\dots,n\}\to\mathbb{R}_+$ through

$$\gamma^\beta(k)=\min\{\Gamma \ : \ \beta(\Gamma,k)\le\epsilon\}. \tag{2.20}$$

Moreover, we focus on functions $\gamma$ that are non-decreasing. When the cost is uncertain, one can again apply either the classical approach from Bertsimas and Sim [2003]

$$\mathcal{U}^{\Gamma}\text{-}CO^{cost} \qquad \min_{x\in\mathcal{X}^{comb}} \max_{u\in\mathcal{U}^{\Gamma}} u^T x,$$

or use the variable uncertainty set

$$\mathcal{U}^{\gamma}\text{-}CO^{cost} \qquad \min_{x\in\mathcal{X}^{comb}} \max_{u\in\mathcal{U}^{\gamma}(x)} u^T x.$$

Let us denote by $z(\mathcal{U}^{\gamma}\text{-}CO^{cost})$ and $z(\mathcal{U}^{\Gamma}\text{-}CO^{cost})$ the optimal solution costs of problems $\mathcal{U}^{\gamma}\text{-}CO^{cost}$ and $\mathcal{U}^{\Gamma}\text{-}CO^{cost}$, respectively. Based on the aforementionned probabilistic results, we can see that both $\mathcal{U}^{\Gamma}\text{-}CO^{cost}$ and $\mathcal{U}^{\gamma}\text{-}CO^{cost}$ provide approximate solutions for combinatorial optimization problems that minimize their Value-at-Risk, which is defined formally as

$$\min_{x\in\mathcal{X}^{comb}} \mathrm{VaR}_{\epsilon}(\tilde{u}^T x),$$

where $\mathrm{VaR}_{\epsilon}(\tilde{u}^T x) = \inf\{t : P(\tilde{u}^T x \leq t) \geq 1-\epsilon\}$, $\tilde{u}_i = \overline{u}_i + \eta_i \hat{u}_i$ is the random variable associated with parameter $u_i$, and $\eta_i$, $i = 1, \ldots, n$, are arbitrary random variables independently and symmetrically distributed in $[-1, 1]$. More precisely, we have the following result.

**Theorem 6.** *Let $\gamma^{\beta}(x)$ be the function defined in (2.9), $\epsilon \in (0, 1)$ be a probability level, and $\Gamma = \gamma^{\beta}(n)$. It holds that*

$$\min_{x\in\mathcal{X}^{comb}} \mathrm{VaR}_{\epsilon}(\tilde{u}^T x) \leq z(\mathcal{U}^{\gamma^{\beta}}\text{-}CO^{cost}) \leq z(\mathcal{U}^{\Gamma}\text{-}CO^{cost}). \tag{2.21}$$

*Proof.* The first inequality follows from (2.7) and the definition of $\gamma^{\beta}$ in (2.20). The second inequality follows from the inclusion $\mathcal{U}^{\gamma^{\beta}}(x) \subseteq \mathcal{U}^{\Gamma}$ for any $x$. □

In view of the above theorem, one can wonder how much cheaper the solution provided by model $\mathcal{U}^{\gamma}$ can be. Unfortunately, we cannot provide a general upper bound for the cost reduction $\frac{z(\mathcal{U}^{\gamma}\text{-}CO^{cost})}{z(\mathcal{U}^{\Gamma}\text{-}CO^{cost})}$. For instance, one readily verifies that the ratio is equal to 1 whenever the optimal solution to $\mathcal{U}^{\Gamma}\text{-}CO^{cost}$, denoted $x^*$, satisfies $\|x^*\|_1 \leq \gamma(\|x^*\|_1)$.

Interestingly, an analytical lower bound can be computed for the ratio. Namely, we show below that when a technical condition holds, the ratio $\frac{z(\mathcal{U}^{\gamma}\text{-}CO^{cost})}{z(\mathcal{U}^{\Gamma}\text{-}CO^{cost})}$ is never smaller that $\frac{\gamma(\lceil\Gamma\rceil)-1}{\lceil\Gamma\rceil}$ and the bound is asymptotically tight with respect to $n$. The next result requires that $\gamma$ satisfies the following condition:

$$\text{there exists a positive integer } \underline{k} \text{ such that } \frac{\gamma(k)-1}{k} \text{ is non-increasing for all } k \geq \underline{k}. \tag{2.22}$$

One can check numerically that function $\gamma^{\beta}$ introduced in Theorem 6 satisfies condition (2.22).

**Theorem 7.** *Let $CO^{cost}$ be any combinatorial optimization problem and $\Gamma = \gamma(n)$. Suppose that $\gamma$ satisfies property (2.22) and that the optimal solution of $\mathcal{U}^{\gamma}$-$CO^{cost}$ has a cardinality greater than or equal to $\underline{k}$. It holds that:*

$$\frac{z(\mathcal{U}^{\gamma}\text{-}CO^{cost})}{z(\mathcal{U}^{\Gamma}\text{-}CO^{cost})} \geq \frac{\gamma(\lceil\Gamma\rceil) - 1}{\lceil\Gamma\rceil}. \tag{2.23}$$

*Bound (2.23) is asymptotically tight with respect to $n$ for $\gamma^{weak}(\|x\|_1) = (-2\ln(\epsilon)\|x\|_1)^{\frac{1}{2}}$, $\overline{u} = 0$, $\hat{u}_i = 1$ for each $i = 1, \ldots, n$, and $\mathcal{X}^{comb} \subseteq \{x, \|x\|_1 = k\}$ for a given integer $k$.*

*Proof.* We first prove that

$$\frac{z(\mathcal{U}^{\gamma}\text{-}CO)}{z(\mathcal{U}^{\Gamma}\text{-}CO)} \geq \frac{\gamma(\lceil\Gamma\rceil) - 1}{\lceil\Gamma\rceil}.$$

Let $x^*$ be the optimal solution of problem $\mathcal{U}^{\gamma}$-$CO$. The following holds:

$$\frac{z(\mathcal{U}^{\gamma}\text{-}CO)}{z(\mathcal{U}^{\Gamma}\text{-}CO)} \geq \frac{\max\limits_{u \in \mathcal{U}^{\gamma}(x^*)} u^T x^*}{\max\limits_{u \in \mathcal{U}^{\Gamma}} u^T x^*} = \frac{\overline{u}^T x^* + \max\limits_{\delta \in [0,1]^n, \sum \delta_i \leq \gamma(\|x^*\|_1)} \sum\limits_i \delta_i \hat{u}_i x_i^*}{\overline{u}^T x^* + \max\limits_{\delta \in [0,1]^n, \sum \delta_i \leq \Gamma} \sum\limits_i \delta_i \hat{u}_i x_i^*} \tag{2.24}$$

$$\geq \frac{\max\limits_{\delta \in [0,1]^n, \sum \delta_i \leq \gamma(\|x^*\|_1)} \sum\limits_i \delta_i \hat{u}_i x_i^*}{\max\limits_{\delta \in [0,1]^n, \sum \delta_i \leq \Gamma} \sum\limits_i \delta_i \hat{u}_i x_i^*} \tag{2.25}$$

$$\geq \frac{\max\limits_{\delta \in [0,1]^n, \sum \delta_i \leq \lfloor\gamma(\|x^*\|_1)\rfloor} \sum\limits_i \delta_i \hat{u}_i x_i^*}{\max\limits_{\delta \in [0,1]^n, \sum \delta_i \leq \lceil\Gamma\rceil} \sum\limits_i \delta_i \hat{u}_i x_i^*} \tag{2.26}$$

$$\geq \frac{\lfloor\gamma(\|x^*\|_1)\rfloor}{\min(\|x^*\|, \lceil\Gamma\rceil)}. \tag{2.27}$$

Inequality (2.24) holds because $x^*$ may not be optimal for $\mathcal{U}^{\Gamma}$-$CO$, inequality (2.25) follows from the fact that $\gamma(\|x^*\|_1) \leq \Gamma$, and inequality (2.27) follows from a more technical argument detailed below.

Because $\lfloor\gamma(\|x^*\|_1)\rfloor \in \mathbb{Z}$ and $\lceil\Gamma\rceil \in \mathbb{Z}$, we can define $\Delta^n \subset \{1, \ldots, n\}$ and $\Delta^d \subset \{1, \ldots, n\}$ as the sets of indices where $\delta_i$ is equal to 1 in the optimal solutions of the maximization problems involved in the numerator and the denominator of (2.26), respectively. Notice that these sets can easily be computed by reordering the items according to the decreasing value of $\hat{u}_i x_i^*$, and taking in $\Delta^n$ (resp. $\Delta^d$) the first $\lfloor\gamma(\|x^*\|_1)\rfloor$ (resp. $\lceil\Gamma\rceil$) elements. Hence, $\lfloor\gamma(\|x^*\|_1)\rfloor \leq \lceil\Gamma\rceil$ implies that that $\Delta^n \subseteq \Delta^d$ so that we can define $\Delta^* = \Delta^d \backslash \Delta^n$. Thus, (2.26) can be rewritten as

$$\frac{\sum\limits_{i \in \Delta^n} \hat{u}_i}{\sum\limits_{i \in \Delta^n} \hat{u}_i + \sum\limits_{i \in \Delta^*} \hat{u}_i}. \tag{2.28}$$

Let us define $\underline{\hat{u}} = \min_{i \in \Delta^n} \hat{u}_i$, which is equal to $\hat{u}_{\lceil \gamma(\|(\|_1 x^*))\rceil}$ according to the reordering just mentioned. By definition, $\hat{u}_i/\underline{\hat{u}} \leq 1$ for each $i \in \Delta^*$. Hence, dividing both members of fraction (2.28) we obtain

$$\frac{\sum_{i \in \Delta^n} \hat{u}_i/\underline{\hat{u}}}{\sum_{i \in \Delta^n} \hat{u}_i/\underline{\hat{u}} + \sum_{i \in \Delta^*} \hat{u}_i/\underline{\hat{u}}} \geq \frac{\sum_{i \in \Delta^n} \hat{u}_i/\underline{\hat{u}}}{\sum_{i \in \Delta^n} \hat{u}_i/\underline{\hat{u}} + \sum_{i \in \Delta^*} 1}. \tag{2.29}$$

Inequality (2.27) finally follows from subtracting $\sum_{i \in \Delta^n}(\hat{u}_i/\underline{\hat{u}} - 1)$ from the two members of the rhs of inequality (2.29) and recalling that $|\Delta^n| = \lfloor \gamma(\|x^*\|_1) \rfloor$ and $|\Delta^n| + |\Delta^*| = \min(\|x^*\|, \lceil \Gamma \rceil)$.

We have proven

$$\frac{z(\mathcal{U}^\gamma\text{-}CO)}{z(\mathcal{U}^\Gamma\text{-}CO)} \geq \frac{\lfloor \gamma(\|x^*\|_1) \rfloor}{\min(\|x^*\|, \lceil \Gamma \rceil)}. \tag{2.30}$$

Two cases are left to analyze to conclude the proof of validity of bound (2.23):

- if $\|x^*\| \geq \lceil \Gamma \rceil$ the rhs of (2.30) becomes $\frac{\lfloor \gamma(\|x^*\|_1) \rfloor}{\lceil \Gamma \rceil}$, which is greater than or equal to $\frac{\lfloor \gamma(\lceil \Gamma \rceil) \rfloor}{\lceil \Gamma \rceil}$ because $\gamma$ is non-decreasing.
- if $\|x^*\| < \lceil \Gamma \rceil$ we obtain

$$\frac{\lfloor \gamma(\|x^*\|_1) \rfloor}{\|x^*\|} \geq \frac{\gamma(\|x^*\|_1) - 1}{\|x^*\|} \geq \frac{\gamma(\lceil \Gamma \rceil) - 1}{\lceil \Gamma \rceil},$$

  where the second inequality follows from (2.22).

We prove next that bound (2.23) is asymptotically tight. Consider optimization problems such that $\overline{u} = 0$, $\hat{u}_i = 1$ for each $i = 1, \ldots, n$, and $\mathcal{X}^{comb} \subseteq \{x, \|x\|_1 = k\}$ for some $k \in \mathbb{Z}_+$ large enough so that (2.22) holds. We see immediately that

$$\frac{z(\mathcal{U}^\gamma\text{-}CO)}{z(\mathcal{U}^\Gamma\text{-}CO)} = \frac{\gamma(k)}{\min(k, \Gamma)} = \max\left(\frac{\gamma(k)}{k}, \frac{\gamma(k)}{\Gamma}\right). \tag{2.31}$$

Let us choose $k$ to minimize the value of (2.31). On the one hand, since $\gamma$ satisfies (2.22), we obtain that $\frac{\gamma(k)}{k}$ is non-increasing for all $k \geq \underline{k}$. On the other hand, $\gamma(k)$ is non-decreasing. Therefore, the minimal value of (2.31) is reached at the value $k \in \mathbb{Z}_+$ where $\frac{\gamma(k)}{k} = \frac{\gamma(k)}{\Gamma}$. That value is $\Gamma$ if $\Gamma$ is integer. Otherwise, the minimum of (2.31) is reached either at $k = \lfloor \Gamma \rfloor$ or at $k = \lceil \Gamma \rceil$, yielding the following value for (2.31):

$$\max\left(\frac{\gamma(\lfloor \Gamma \rfloor)}{\lfloor \Gamma \rfloor}, \frac{\gamma(\lceil \Gamma \rceil)}{\Gamma}\right), \tag{2.32}$$

which is not smaller than bound (2.23). Consider next the particular function $\gamma(x) = (-2\ln(\epsilon)\|x\|)^{\frac{1}{2}}$ and denote $(-2\ln(\epsilon))^{\frac{1}{2}}$ by $K > 0$ and $\Gamma = \gamma(n) = (-2\ln(\epsilon)n)^{\frac{1}{2}} = Kn^{\frac{1}{2}}$ by

$m^{\frac{1}{2}}$. We show below that the value of (2.32) converges to the rhs of (2.23) as $m = K^2 n$ goes to infinity:

$$\begin{aligned}\max\left(\frac{K(\lfloor m^{\frac{1}{2}}\rfloor)^{\frac{1}{2}}}{\lfloor m^{\frac{1}{2}}\rfloor}, \frac{K(\lceil m^{\frac{1}{2}}\rceil)^{\frac{1}{2}}}{m^{\frac{1}{2}}}\right) - \frac{K(\lceil m^{\frac{1}{2}}\rceil)^{\frac{1}{2}} - 1}{\lceil m^{\frac{1}{2}}\rceil} &\leq \frac{K(\lceil m^{\frac{1}{2}}\rceil)^{\frac{1}{2}}}{\lfloor m^{\frac{1}{2}}\rfloor} - \frac{K(\lceil m^{\frac{1}{2}}\rceil)^{\frac{1}{2}} - 1}{\lceil m^{\frac{1}{2}}\rceil} \\ &\leq \frac{K(\lceil m^{\frac{1}{2}}\rceil)^{\frac{1}{2}} + 1}{m^{\frac{1}{2}} + 1} - \frac{K(\lceil m^{\frac{1}{2}}\rceil)^{\frac{1}{2}} - 1}{m^{\frac{1}{2}} + 1} \\ &\leq \frac{2}{m^{\frac{1}{2}} + 1}.\end{aligned}$$

□

Function $\gamma^{weak}$ mentioned in Theorem 7 is related to the function $\Gamma^{weak}$ introduced in the previous chapter.

## 2.6 Numerical experiments

We assess in this section the numerical efficiency of the model $\mathcal{U}^{\partial\gamma^\beta}$, where $\partial\gamma^\beta$ is the best affine over-approximation of $\gamma^\beta$, see Poss [2014] for details.

### 2.6.1 Knapsack problem

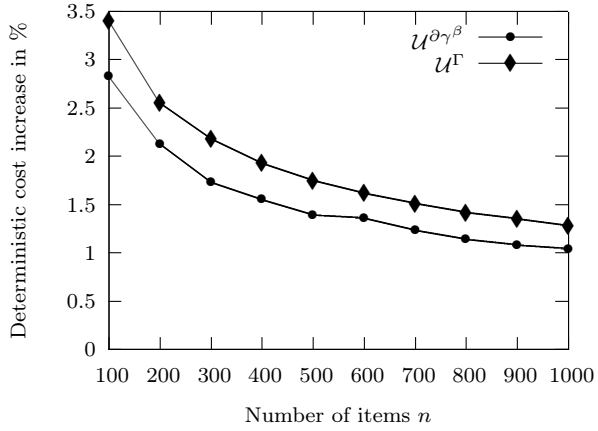
Given a set of $n$ items, each with profit $p_i$ and weight $u_i$, the knapsack problem aims at choosing a subset of these items not exceeding the available capacity $b$ and maximizing the profit:

$$KP \qquad \max\left\{\sum_{i=1}^n p_i x_i \ : \sum_{i=1}^n u_i x_i \leq b, x \in \{0,1\}^n\right\}. \tag{2.33}$$

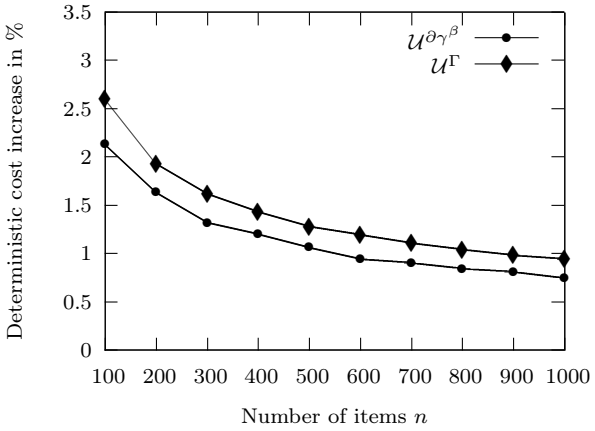
The problem has been used by Bertsimas and Sim [2004] to evaluate the cost of protecting the capacity constraint for various probability guarantees.

We generate our instances as the one used by Bertsimas and Sim [2004]. We consider different item numbers $n \in \{100, 200, \ldots, 1000\}$ and set the capacity to $b = 20n$ for each value of $n$. For each value of $n$, we generate randomly five instances as follows. For each $i = 1, \ldots, n$, the average weight $\overline{u}_i$ is chosen uniformly from the set $\{20, 21, \ldots, 29\}$, the deviation $\hat{u}_i$ is equal to 10% of $\overline{u}_i$, and the profit $p_i$ is chosen uniformly from the set $\{16, 17, \ldots, 77\}$.

We compare in Figure 2.1 the optimal protection costs (prices of robustness) of the different models. Let $z(KP)$, $z(\mathcal{U}^\Gamma\text{-}KP)$, and $z(\mathcal{U}^{\partial\gamma^\beta}\text{-}KP)$ denote the optimal solution costs to, respectively, the deterministic model and the robust models where the knapsack constraint must be satisfied for all values of $u$ in $\mathcal{U}^\Gamma$ or $\mathcal{U}^{\partial\gamma^\beta}$. We compute the cost $c(*)$




(a) $\epsilon = 0.01$




(b) $\epsilon = 0.05$

Figure 2.1: Price of robustness for the knapsack problem.

of protecting a solution with a given probability for model $*$ as $c(*) = \frac{z(KP)-z(*)}{z(KP)}$. We present in Figure 2.1 the geometric means of these protection costs for each value of $n$. On average, $c(\mathcal{U}^{\partial\gamma^\beta}\text{-}KP)$ is 18% less than $c(\mathcal{U}^\Gamma\text{-}KP)$.

Notice that the price of robustness was introduced by Bertsimas and Sim [2004] and we use that measure here to be as close as possible to the results obtained by Bertsimas and Sim [2004]. In practical applications however, we argue that it makes more sense to study the *Benefit of Robustness* because decision makers tend to overestimate the uncertain parameters in the presence of uncertainty. We come back to this point in the next chapter.

We compare then the computational complexity of the dualized formulations obtained for models $\mathcal{U}^{\partial\gamma^\beta}$ and $\mathcal{U}^\Gamma$. Let $t(*)$ be the solution time in seconds to solve model $*$ to optimality. The solution time was less than 10 second for any of our instances. For $\epsilon = 0.01$, the geometric mean of the ratios $t(\mathcal{U}^{\partial\gamma^\beta}\text{-}KP)/t(\mathcal{U}^\Gamma\text{-}KP)$ is equal to 1.7, with a maximum value of 7.68. For $\epsilon = 0.05$, these values increase to 2.5 and 10, respectively.

The ratios do not increase with the problem size. In addition to the solution times, we investigate the bound provided by the linear relaxation of the problem. In particular, we want to understand whether the bound proposed in Theorem 4 for the big-$M$ coefficients is tight. Our results show that the gap between the linear relaxation and the solution of the problem are very close for both models. Surprisingly, the LP relaxation gap of problem $\mathcal{U}^{\partial\gamma^\beta}$-$KP$ is even 10% better in average than the one of $\mathcal{U}^{\Gamma}$-$KP$. However, replacing $\max_j(\hat{u}_j)$ by very large numbers reduce significantly the bound provided by the linear relaxation. For instance, setting $M$ to 10000 multiplies the gap by an average factor of four.

We have also tested more refined linearizations, using two and three linear over-approximations. Unreported results have shown that the solution times tend to increase more than linearly with the number of linear functions used while decreasing the protecting cost by less than 1%.

### 2.6.2 Shortest path problem

We illustrate next the cost-reduction obtained with model $\mathcal{U}^{\partial\gamma^\beta}$ for the robust shortest path problem ($SP$). Using Theorem 5, we can solve the $\mathcal{U}^{\partial\gamma^\beta}$-$SP$ in essentially the same amount of time as for $\mathcal{U}^{\Gamma}$-$SP$. The cost reduction is illustrated in Figure 2.2, using the real road networks described in Table 2.1.

| Network name | | | | Arc Length | | |
|---|---|---|---|---|---|---|
| (abbreviation) | $\|V\|$ | $\|A\|$ | Arc/Node ratio | Maximum | Mean | Stnd. Dev. |
| Nebraska (NE1) | 523 | 1646 | 3.14 | 0.874764 | 0.215551 | 0.142461 |
| Alabama (AL1) | 842 | 2506 | 2.98 | 0.650305 | 0.128870 | 0.114031 |
| Minnesota (MN1) | 951 | 2932 | 3.08 | 0.972436 | 0.175173 | 0.132083 |
| Iowa (IA1) | 1003 | 2684 | 2.68 | 0.573768 | 0.119900 | 0.113719 |

Table 2.1: Characteristics of the networks taken from Zhan and Noon [1998].

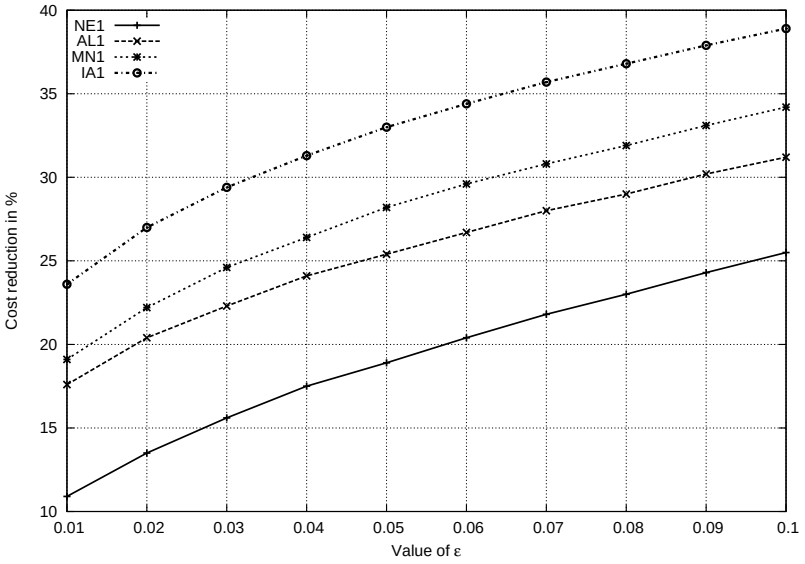



Figure 2.2: Cost reduction when using uncertainty model $\mathcal{U}^{\partial\beta}$-$CO$ instead of uncertainty model $\mathcal{U}^{\Gamma}$-$CO$.

# Chapter 3

# Constrained shortest path

*The chapter is based on Pessoa et al. [2015].*

## 3.1 Introduction

Let $G = (N, A)$ be a directed graph with two special nodes $o$ and $d$, $\kappa : A \to \mathbb{R}_+$ be a cost function and $u : A \to \mathbb{Z}_+$ be a weight function that represents the resource consumption along each arc of $G$. The resource constrained shortest path problem looks for the shortest path (according to cost $\kappa$) from $o$ to $d$ that satisfies the resource constraint(s), which can be, for instance, capacity constraints or time windows. Namely, a capacity constraint considers an upper bound $W \in \mathbb{Z}_+$ on the resource available on a path, and imposes that any feasible path $p$ satisfies

$$w(p) \leq W, \tag{3.1}$$

where $w(p) = \sum_{a \in p} u_a$. The shortest path problem with capacity constraint is denoted by $CSP$. Different from the capacity constraint, time windows must be satisfied at each node pertaining to feasible paths and they are defined by two vectors $\underline{b}$ and $\overline{b}$ in $\mathbb{Z}_+^{|N|}$ such that $\underline{b}_i \leq \overline{b}_i$ for each $i \in N$. Consider a path $p = (i_0, i_1, \ldots, i_l)$ containing $l = |p|$ arcs. For each $j \in \{1, \ldots, l\}$, we define the arrival time at node $i_j$ as $t_j(p) = \max(\underline{b}_{i_j}, t_{j-1}(p) + u_{i_{j-1}i_j})$, and the time window constraints are written as

$$t_j(p) \leq \overline{b}_{i_j}, \quad j = 1, \ldots, l. \tag{3.2}$$

The shortest path problem with time windows is denoted by $TWSP$. One can define similarly problems with multiple capacity constraints and/or time window constraints. To keep simple notations, we disregard these generalizations in the following and consider a unique resource constraint.

In what follows, we suppose that the weights are uncertain parameters that can take any value in a given uncertainty set $\mathcal{U} \subset \mathbb{Z}_+^{|A|}$, which we assume to be finite. Given a fixed $u \in \mathcal{U}$, we redefine the values of $w(p)$ and $t_j(p)$ as $w(p, u)$ and $t_j(p, u)$, respectively, to

mark the dependency on the uncertain parameter $u$. In the uncertain context, we impose that constraints (3.1) and (3.2) be satisfied for all $u \in \mathcal{U}$; that is

$$w(p,u) \leq W, \quad u \in \mathcal{U}, \tag{3.3}$$

and

$$t_j(p,u) \leq \bar{b}_{i_j}, \quad j = 1,\ldots,l,\ u \in \mathcal{U}. \tag{3.4}$$

We denote the associated optimization problems by $\mathcal{U}$-$CSP$ and $\mathcal{U}$-$TWSP$, respectively.

We show in Section 3.2 that the problems are $\mathcal{NP}$-hard in the strong sense for uncertainty sets of unbounded cardinalities. In Section 3.3, we solve $\mathcal{U}^\Gamma$-$CSP$ by solving $|A|+1$ deterministic $CSP$ problems by applying Theorem 2 and Theorem 5. We show then in Section 3.4 that $\mathcal{U}^\Gamma$-$TWSP$ is $\mathcal{NP}$-hard in the strong sense, which highlights a fundamental difference between $\mathcal{U}^\Gamma$-$CSP$ and $\mathcal{U}^\Gamma$-$TWSP$. We present in Section 3.5 a label-setting algorithm based on that proposed by Desrochers and Soumis [1988], paying a particular attention to the generalization of dominance rules.

## 3.2 $\mathcal{NP}$-hardness for general uncertainty sets

When $\mathcal{U}$ is reduced to a singleton, $\mathcal{U}$-$CSP$ and $\mathcal{U}$-$TWSP$ reduce to their deterministic counterparts. Otherwise, the problems can be seen as deterministic problems that contain multiple capacity or time windows constraints, one for each value of $u$ in $\mathcal{U}$. When the cardinality of $\mathcal{U}$ is bounded by a small constant, $\mathcal{U}$-$CSP$ and $\mathcal{U}$-$TWSP$ could therefore be solved in pseudo-polynomial time using, for instance, the classical label-setting algorithm. However, the running time of such algorithms is exponential in the cardinality of $\mathcal{U}$, making them impractical for solving problems with uncertainty sets of large cardinalities. One can wonder whether different efficient approaches could exist to solve these problems under arbitrary uncertainty sets. The next result answers this in the negative, showing that $\mathcal{U}$-$CSP$ and $\mathcal{U}$-$TWSP$ are $\mathcal{NP}$-hard in the strong sense when the cardinality of $\mathcal{U}$ is unbounded. The proof, omitted for brevity, simply consists in reducing the robust knapsack problem ($\mathcal{U}$-$KP$) to $\mathcal{U}$-$CSP$. A similar proof can be devised to reduce $\mathcal{U}$-$KP$ to $\mathcal{U}$-$TWSP$.

ROBUST KNAPSACK PROBLEM ($\mathcal{U}$-$KP$)

Input: Set $\{1,\ldots,m\}$ of items, set $\mathcal{V} \subset \mathbb{Z}_+^m$ of weights $w$, profit vector $c \in \mathbb{R}_+^m$, capacity $D$.

Task: Find a subset of items $I \subseteq \{1,\ldots,m\}$ of maximum profit such that $\sum_{i \in I} w_i \leq D$ for all $w \in \mathcal{V}$.

**Theorem 8.** *$\mathcal{U}$-$CSP$ is $\mathcal{NP}$-hard in the strong sense for an uncertainty set $\mathcal{U}$ of unbounded cardinality.*

The above result suggests that problems $\mathcal{U}$-$CSP$ and $\mathcal{U}$-$TWSP$ can be quite hard to solve exactly for arbitrary uncertainty sets. This is not very surprising as it is well known that many robust versions of combinatorial optimization problems become $\mathcal{NP}$-hard in the presence of uncertainty [Kouvelis and Yu, 2013]. In view of the above difficulty, we focus on uncertainty sets $\mathcal{U}^\Gamma$ and $\mathcal{U}^\gamma$, known to often lead to robust optimization problems with computational complexities similar to the complexities of their deterministic counterparts.

## 3.3 Complexity of $\mathcal{U}^\Gamma$-$CSP$ and $\mathcal{U}^\gamma$-$CSP$

We consider in this section $\mathcal{U}^\gamma$-$CSP$ and show how the problem can be solved in pseudo-polynomial time. Let us formulate next $\mathcal{U}^\gamma$-$CSP$ as an integer linear program with one linear robust constraint. Let $x \in \{0,1\}^{|A|}$ be a vector of optimization variables stating which arcs belong to the solution, and let $\mathcal{X}^{comb} \subset \{0,1\}^{|A|}$ contain all vectors $x$ that correspond to paths from $o$ to $d$. With these variables, $\mathcal{U}^\gamma$-$CSP$ can be cast as

$$\min \left\{ \sum_{a \in A} \kappa_a x_a : \sum_{a \in A} u_a x_a \leq W, u \in \mathcal{U}^\gamma(x), x \in \mathcal{X}^{comb} \right\}. \tag{3.5}$$

Applying Theorem 5 to problem (3.5) yields the results below.

**Corollary 1.** *If $\gamma$ is affine, $\mathcal{U}^\gamma$-$CSP$ can be handled by solving $|A|+1$ problems $CSP$ with modified weights.*

The above result implies that $\mathcal{U}^\gamma$-$CSP$ can be solved in pseudo-polynomial time. When the appropriate data structures are used, $CSP$ can be solved by the label-setting algorithm in $O(|A|W)$ if there are no zero weights [Desrochers and Soumis, 1988], yielding the following corollary.

**Corollary 2.** *If $\gamma$ is affine, $\mathcal{U}^\gamma$-$CSP$ can be solved in $O(|A|^2 W)$ time.*

Notice that whenever $\gamma$ is not affine, we can still solve $\mathcal{U}^\gamma$-$CSP$ in pseudo-polynomial time, albeit at a higher computational cost. Let $CSP_k^{\leq}$ and $\mathcal{U}^\Gamma$-$CSP_k^{\leq}$ be defined as $CSP$ and $\mathcal{U}^\Gamma$-$CSP$, respectively, with the additional restriction that feasible paths cannot contain more than $k$ arcs, for $0 \leq k \leq |N|$, and $\Gamma = \gamma(k)$. Clearly, the label-setting algorithm applied to $CSP_k^{\leq}$ with the appropriate data structures has a running time of $O(|A|W|N|)$. Moreover, Theorem 5 can be applied to $\mathcal{U}^\Gamma$-$CSP_k^{\leq}$ so that the problem can be solved in $O(|A|^2 W|N|)$ time.

**Theorem 9.** *$\mathcal{U}^\gamma$-$CSP$ can be solved in $O(|A|^2 W|N|)$ time for any non-decreasing function $\gamma$.*

Unfortunately, the approach used in this section cannot be used to provide pseudo-polynomial algorithms for $\mathcal{U}^\gamma$-$TWSP$. The reason is that time windows restrictions are

expressed through $|N|$ non-linear constraints, instead of a single linear constraint as is the case for $\mathcal{U}^{\gamma}$-$CSP$. An alternative would be to adapt the layered formulation from Agra et al. [2012] to $\mathcal{U}^{\gamma}$-$TWSP$. Nevertheless, the resulting formulation would contain many robust constraints (more than $|N|^2$), so that applying Theorem 5 to it would enable us to solve $\mathcal{U}^{\Gamma}$-$TWSP$ by solving more than $O\left(|A|^{|N|^2}\right)$ problems $TWSP$, instead of the $O(|A|)$ problems involved in Corollary 1. This approach could hardly be of any practical use.

## 3.4 Complexity of $\mathcal{U}^{\Gamma}$-$TWSP$ and $\mathcal{U}^{\gamma}$-$TWSP$

We show in this section that the decision version of a simplification of $\mathcal{U}^{\Gamma}$-$TWSP$ is $\mathcal{NP}$-complete in the strong sense. Namely, we consider the robust path with deadlines, obtained from the decision version of $\mathcal{U}^{\Gamma}$-$TWSP$ by looking for a path with zero nominal travel time that satisfies the upper time windows constraints (hence we suppose $\overline{u} = 0$ and $\underline{b} = 0$); we also suppose that the graph is acyclic.

ROBUST PATH WITH DEADLINES ($\mathcal{U}^{\Gamma}$-$PD$)

Input: A directed acyclic graph $D = (N, A)$ with corresponding $\hat{u}_a$ for each $a \in A$ and $\overline{b}_i$ for each $i \in N$, and a positive integer $\Gamma$.

Question: Does there exist a path $p = (i_0, i_1, \ldots, i_l)$ in $D$ starting at $o \in N$ ($i_0 = o$) and ending at $d \in N$ ($i_l = d$) such that

$$\max\left\{\sum_{a\in S}\hat{u}_a \ : S \subseteq (i_0, i_1, \ldots, i_h), |S| \leq \Gamma\right\} \leq \overline{b}_{i_h}, \text{ for each } h = 1, \ldots, l?$$

Our proof is based on reducing the decision version of the independent set problem to $\mathcal{U}^{\Gamma}$-$PD$. It is well known that the former problem is $\mathcal{NP}$-complete in the strong sense (e.g., Garey and Johnson [1990]).

INDEPENDENT SET ($IS$)

Input: An undirected graph $G = (V, E)$ and a positive integer $K$.

Question: Does there exist $W \subseteq V$ such that $|W| \geq K$ and $\{i, j\} \nsubseteq W$ for each $\{i, j\} \in E$ ?

**Theorem 10.** *$\mathcal{U}^{\Gamma}$-$PD$ is $\mathcal{NP}$-complete in the strong sense.*

*Proof.* First, we show that $\mathcal{U}^{\Gamma}$-$PD$ belongs to $\mathcal{NP}$. For that, the feasibility of a path $p = (i_0, i_1, \ldots, i_l)$ must be checked in polynomial time. This is true because, for each $h = 1, \ldots, l$, the time window constraint on $i_h$ needs to be checked only for a single $S$

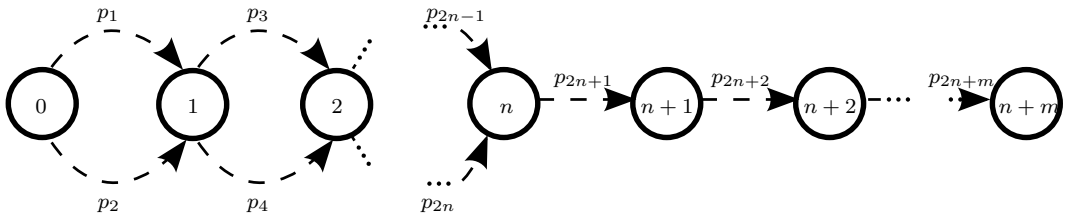



Figure 3.1: Reduction from the independent set problem to $\mathcal{U}^{\Gamma}$-$PD$.

containing $\min(h, \Gamma)$ arcs, with largest values of $\hat{u}_a$. Next, we show that $IS$ can be reduced to $\mathcal{U}^{\Gamma}$-$PD$.

Given an instance to $IS$ with $|V| = n$ nodes and $|E| = m$ edges, we show next how to build an instance for $\mathcal{U}^{\Gamma}$-$PD$. The graph $D$ is described on Figure 3.1 where $p_i$ represents a directed path that contains $m+1$ arcs for each $i \in \{1, \ldots, 2n\}$ and $n$ arcs for each $i \in \{2n+1, \ldots, 2n+m\}$. Moreover, $o = 0$, $d = n+m$ and $\Gamma = n(m+1)$. In what follows we denote the $k$-th arc of $p_i$ by $a(i,k)$ and the elements of $E$ by $\{e_1, \ldots, e_m\}$.

One sees immediately that any path $p$ from $o$ to $d$ must contain all paths $p_{2n+i}$ for $i \in \{1, \ldots, m\}$. Furthermore, for each $i \in \{1, \ldots, n\}$, the path $p$ contains either $p_{2i}$ or $p_{2i-1}$. Hence, there are $2^n$ different paths in $D$ from $o$ to $d$, which is as many as the number of different subsets of $V$. The correspondence between subsets of $V$ and paths in $D$ works as follows. Let $W \subseteq V$ be a subset of $V$ and let $p_W$ be the path in $D$ associated to $W$. Then, for each $i \in \{1, \ldots, n\}$, the path $p_W$ contains $p_{2i}$ if $i \in W$ and the path $p_W$ contains $p_{2i-1}$ otherwise. We can describe concisely $p_W$ by introducing the function $\widetilde{W}$ defined as follows: $\widetilde{W}(i) = 2i$ if $i \in W$ and $\widetilde{W}(i) = 2i-1$ otherwise, for each $i \in \{1, \ldots, n\}$. Hence, $p_W = p_{\widetilde{W}(1)} \cup p_{\widetilde{W}(2)} \cup \cdots \cup p_{\widetilde{W}(n)} \cup p_{2n+1} \cup \cdots \cup p_{2n+m}$.

We explain next how to choose the parameters $\hat{u}$ and $\bar{b}$ such that the deadline constraints imposed on $p_W$ are equivalent to the constraints of $IS$ imposed on $W$. Namely, we choose these parameters such that the constraint

$$\max\left\{\sum_{a\in S} \hat{u}_a \ : S \subseteq p_{\widetilde{W}(1)} \cup p_{\widetilde{W}(2)} \cup \cdots \cup p_{\widetilde{W}(n)}, |S| = n(m+1)\right\} \leq \bar{b}_n \tag{3.6}$$

written for $p_W$ is equivalent to $|W| \geq K$ and constraints

$$\max\left\{\sum_{a\in S} \hat{u}_a \ : S \subseteq p_{\widetilde{W}(1)} \cup \cdots \cup p_{\widetilde{W}(n)} \cup p_{2n+1} \cup \cdots \cup p_{2n+h}, |S| = n(m+1)\right\} \leq \bar{b}_{n+h} \tag{3.7}$$

written for $p_W$ are equivalent to $e_h = \{i, j\} \nsubseteq W$, for each $h = 1, \ldots, m$. Notice that in (3.6) and (3.7), $|S| = n(m+1)$ has been used instead of $|S| \leq n(m+1)$ because all components of $\hat{u}$ are positive and the paths considered in (3.6) and (3.7) contain not

less than $n(m+1)$ arcs. Our construction below is made in two steps. First, we impose restrictions on $\hat{u}$ and $\overline{b}$ which ensure that the corresponding deadline constraints translate exactly into the constraints of $IS$. To this end we introduce additional parameters $\mu, \nu$, and $\sigma_h$ for each $h \in \{1, \ldots, m\}$ to be specified later. Second, we provide an example of a vector $(\mu, \nu, \sigma, \hat{u}, \overline{b})$ that satisfies the restrictions and that is composed of polynomial functions of $n, m$, and $K$. For brevity, the construction of that vector is not presented in this manuscript, we refer to Pessoa et al. [2015] for details.

**Constraint** (3.6) The number of arcs of the subpath $p_{\widetilde{W}(1)} \cup p_{\widetilde{W}(2)} \cup \cdots \cup p_{\widetilde{W}(n)}$ is equal to $n(m+1)$, so that the unique choice of $S$ in the maximization is the full subpath $p_{\widetilde{W}(1)} \cup p_{\widetilde{W}(2)} \cup \cdots \cup p_{\widetilde{W}(n)}$. Hence, (3.6) becomes

$$\sum_{i=1}^{n} \sum_{k=1}^{m+1} \hat{u}_{a(\widetilde{W}(i),k)} \leq \overline{b}_n. \tag{3.8}$$

Let $\hat{u}$ and $\overline{b}_n$ be such that

$$\begin{aligned} \sum_{k=1}^{m+1} \hat{u}_{a(2i,k)} &= \mu, & i \in \{1, \ldots, n\}, \\ \sum_{k=1}^{m+1} \hat{u}_{a(2i-1,k)} &= \mu + 1, & i \in \{1, \ldots, n\}, \\ \overline{b}_n &= n\mu + n - K. \end{aligned} \tag{3.9}$$

Plugging (3.9) into (3.8), we obtain that $p_W$ cannot contain more than $n - K$ paths of the form $p_{2i-1}$ for $i \in \{1, \ldots, n\}$, which is equivalent to $|V \setminus W| \leq n - K$, or more simply, $|W| \geq K$.

**$h$-th constraint of** (3.7) We next propose restrictions on $\hat{u}$ and $\overline{b}$ such that the maximization in the lhs of the $h$-th constraint in (3.7) becomes

$$\sum_{i=1}^{n} \sum_{k=h+1}^{m+1} \hat{u}_{a(\widetilde{W}(i),k)} + \sum_{i=2n+1}^{2n+h} \sum_{k=1}^{n} \hat{u}_{a(i,k)}. \tag{3.10}$$

One readily checks that the following set of constraints yields the desired (3.10):

$$\begin{aligned} \hat{u}_{a(i,k)} &< \hat{u}_{a(j,k+1)}, & i, j \in \{1, \ldots, 2n\}, k \in \{1, \ldots, m\}, \\ \nu &> \hat{u}_{a(i,m)}, & i \in \{1, \ldots, 2n\}, \\ \nu &= \hat{u}_{a(i,k)}, & i \in \{2n+1, \ldots, 2n+m\}, k \in \{1, \ldots, n\}. \end{aligned} \tag{3.11}$$

Once the lhs of the $h$-th constraint of (3.7) has been turned into (3.10), we impose the following additional restrictions to obtain the desired deadline constraint:

$$\begin{aligned}
\sum_{k=h+1}^{m+1} \hat{u}_{a(2i,k)} &= \begin{cases} \sigma_h + 1, & \text{if } e_h \text{ is adjacent to the node } i, \\ \sigma_h, & \text{otherwise.} \end{cases}, \quad i \in \{1, \ldots, n\}, \\
\sum_{k=h+1}^{m+1} \hat{u}_{a(2i-1,k)} &= \sigma_h, \qquad\qquad i \in \{1, \ldots, n\}, \\
\overline{b}_{n+k} &= 1 + n\sigma_h + hn\nu.
\end{aligned} \tag{3.12}$$

Plugging restrictions (3.11) and (3.12) into the $h$-th constraint of (3.7), we obtain that $p_W$ cannot contain more than a single subpath $p_{2i}$ such that $i$ is adjacent to edge $e_h$. Hence, the number of vertices in $W$ adjacent to $e_h$ must not be greater than 1, which is equivalent to $\{i, j\} \nsubseteq W$ for $e_h = \{i, j\}$. □

Since $\mathcal{U}^\Gamma$-$PD$ is a special case of the decision problem associated to $\mathcal{U}^\Gamma$-$TWSP$, we obtain immediately the following result, which contrasts with the pseudo-polynomial algorithms proposed for $\mathcal{U}^\Gamma$-$CSP$ in the previous section.

**Corollary 3.** *$\mathcal{U}^\Gamma$-$TWSP$ and $\mathcal{U}^\gamma$-$TWSP$ are $\mathcal{NP}$-hard in the strong sense.*

## 3.5 Label-setting algorithm

We show in this section how to extend the classical labels used in dynamic programming and label-setting algorithms (e.g., Desrochers and Soumis [1988]) to the case of uncertain weights. We focus on the definition of robust labels and discuss dominance rules in case of budgeted uncertainty. Our description is presented for time windows; one can readily modify the algorithms presented below for a capacity constraint instead of time windows. Notice, however, that the computational complexity of the algorithm presented below is exponential in $\Gamma$, which is in accordance with the $\mathcal{NP}$-hardness of $\mathcal{U}^\Gamma$-$TWSP$.

In Subection 3.5.1, we sketch the label-setting algorithm, recall how labels are defined in the deterministic setting and how dominance applies. We refer the interested reader to Boland et al. [2006], among others, for a detailed description of the algorithm. In Subection 3.5.2, we consider the set $\mathcal{U}^\Gamma$ and suppose that $\Gamma$ is integer. If it is not the case, we can always round up $\Gamma$ to obtain a slightly more conservative model. In the (unlikely) situation where the fractional part of $\Gamma$ really matters, one can always extend the algorithm described next in a way similar to the dynamic programming algorithms described by Poss [2014]. The extension to variable $\mathcal{U}^\gamma$ is omitted for brevity.

### 3.5.1 Deterministic labels

We outline next the basic principles of the label-setting algorithm for $TWSP$. For each path $p$ from the node $o$ to the node $i$, the classical label-setting algorithm considers the label $(\kappa(p), t_{|p|}(p))$ that records the cost of the path and its arrival time at the node $i$. To avoid constructing all labels, the algorithm selects and extends labels in a special order. Two sets of labels are considered: permanent labels and non-permanent labels. At each iteration, the algorithm selects the smallest non-permanent label according to the lexicographical order and marks the label as permanent. Then, for each direct descendant of the node associated to the current label, we create a new non-permanent label and check whether it satisfies the time-windows and is not dominated by a permanent label. If one of the condition fails to hold, the label is immediately removed from the set of non-permanent labels. The algorithm ends when the next selected label corresponds to $d$.

A crucial phase in the label-setting algorithm is the removal of dominated labels, which reduces significantly the total number of labels searched in the course of the algorithm. Given two labels $y$ and $y'$ associated to paths $p$ and $p'$ ending at the same node, we say that the label $y'$ is dominated by the label $y$ if the following condition holds: *if path $p'$ belongs to an optimal solution of $TWSP$, then the path $p$ belongs to an optimal solution of $TWSP$.* Dominated labels can be discarded from the search that occurs during the label-setting algorithm. The next result is well known and is presented without proof.

**Lemma 2.** *Consider the $TWSP$ and let $y = (\kappa, t)$ and $y' = (\kappa', t')$ be two labels associated to paths $p$ and $p'$ ending at the same node. Assume the following conditions hold:*

1. $\kappa \leq \kappa'$
2. $t \leq t'$
3. *and at least one inequality is strict.*

*Then, label $y'$ is dominated by label $y$.*

### 3.5.2 Robust labels

Next, we consider the robust time windows constraints (3.4) recalled below for uncertainty set $\mathcal{U}^\Gamma$

$$t_j(p, u) \leq \overline{b}_{i_j}, \quad j = 1, \ldots, l, \ u \in \mathcal{U}^\Gamma, \tag{3.13}$$

where $p = (i_0, i_1, \ldots, i_l)$, and $l = |p|$. The naive approach to $\mathcal{U}^\Gamma$-$TWSP$ would express the problem as a shortest path problem with $|\mathcal{U}^\Gamma|$ time-windows constraints, one for each $u \in \mathcal{U}^\Gamma$. Defining $s = |\mathcal{U}^\Gamma|$ and $\mathcal{U}^\Gamma = \{u^1, \ldots, u^s\}$, the naive approach would associate to the node $i$ and each path $p$ from $o$ to $i$ a label with $s$ resources,

$$\left(\kappa(p), t_{|p|}(p, u^1), \ldots, t_{|p|}(p, u^s)\right), \tag{3.14}$$

and extend the label through arc $(i, j)$ with the classical formula

$$\big(\kappa(p) + \kappa_{ij}, \max(\underline{b}_i, t_{|p|}(p, u^1) + u^1_{ij}), \ldots, \max(\underline{b}_i, t_{|p|}(p, u^s) + u^s_{ij})\big). \tag{3.15}$$

Using labels (3.14) and appropriate data structures, one can solve $\mathcal{U}^\Gamma$-$TWSP$ by the label-setting algorithm in time $O(s|A|B^s)$, where $s = \sum_{k=1}^{\Gamma} \binom{|A|}{k}$. The main result of this section is to show how to reduce the solution time to $O(\Gamma|A|B^{\Gamma+1})$ by using a more compact description of the labels. Notice that this is a significant improvement since $\Gamma$ is always much smaller than $s$. To be more precise, this improvement leads to pseudo-polynomial algorithms when $\Gamma$ is fixed, which happens in applications where we are interested in being protected only against a small number of deviations, regardless to the size of the instances. However, if we choose $\Gamma$ according to probabilistic guarantees, then $\Gamma \in O(|A|^{1/2})$ (see the weak analytical bound $\Gamma^{weak}$ presented after (1.12)). In the latter case, the resulting computing time of the label-setting algorithm would be $O(|A|^{3/2}B^{O(|A|^{1/2})})$ which is not polynomial, although asymptotically smaller than $O(s|A|B^s)$.

Before explaining how the running time can be decreased to $O(\Gamma|A|B^{\Gamma+1})$, we present without proof the well known extension of Lemma 2 to multiple-resource label (3.14).

**Lemma 3.** *Consider the $\mathcal{U}^\Gamma$-$TWSP$ and let $y = (\kappa, t^1, \ldots, t^s)$ and $y' = (\kappa', t'^1, \ldots, t'^s)$ be two labels associated to paths $p$ and $p'$ ending at the same node. Assume the following conditions hold:*

1. *$\kappa \leq \kappa'$*
2. *$t^j \leq t'^j$, for each $j = 1, \ldots, s$*
3. *and at least one inequality is strict.*

*Then, label $y'$ is dominated by label $y$.*

The key idea to reduce the running time of the robust label-setting algorithm is based on rewriting the time windows constraints (3.13) as

$$\max_{u \in \mathcal{U}^\Gamma} t_j(p, u) \leq \overline{b}_{i_j}, \quad j = 1, \ldots, l. \tag{3.16}$$

Our objective is to define *robust labels* that contain only the necessary information to test whether the current path is feasible when considering its maximum travel time over $\mathcal{U}^\Gamma$. Rather than considering all possible travel times $u \in \mathcal{U}^\Gamma$ that could be used along the path $p = (i_0, \ldots, i_l)$ from $o = i_0$ to $i = i_l$, we can define the label attached to the node $i$ and the path $p$ as

$$\left(\kappa(p), \tau^0_l(p), \ldots, \tau^\Gamma_l(p)\right), \tag{3.17}$$

where, for each $j = 1, \ldots, l$, $\tau^g_j(p)$ is defined as the maximum arrival time at node $i_j$ when considering up to $g \in \{0, \ldots, \Gamma\}$ deviations when $g \leq |p|$ and is equal 0 otherwise; that

is,

$$\tau_j^g(p) = \begin{cases} \max\limits_{u \in \mathcal{U}^g} t_j(p,u), & \text{for each } g \in \{0,\dots,\min(|p|,\Gamma)\}, \\ 0, & \text{for each } g \in \{|p|+1,\dots,\Gamma\}. \end{cases} \tag{3.18}$$

Then, we extend the label through arc $(i,k)$, generating a new label for node $k = i_{l+1}$, by the formula

$$\begin{cases} \kappa = \kappa(p) + \kappa_{ij}, \\ \tau_{l+1}^0 = \max\left(\underline{b}_j, \tau_l^0(p) + \overline{u}_{ij}\right), \\ \tau_{l+1}^g = \max\left(\underline{b}_j, \tau_l^{g-1}(p) + \overline{u}_{ij} + \hat{u}_{ij}, \tau_l^g(p) + \overline{u}_{ij}\right), \quad \text{for each } g \in \{1,\dots,\bar{g}\}, \\ \tau_{l+1}^{\bar{g}+1} = \max\left(\underline{b}_j, \tau_l^{\bar{g}-1}(p) + \overline{u}_{ij} + \hat{u}_{ij}\right), \end{cases} \tag{3.19}$$

where $\bar{g} = \min\left(|p|, \Gamma - 1\right)$ and the extended label is feasible if $\tau^{\bar{g}+1}$ is less than or equal to $\overline{b}_j$. It is easy to see by induction that extending the label $(0,0,\dots,0)$, that corresponds to the empty path, iteratively through formula (3.19) leads exactly to definition (3.18). One readily sees that the solution time of the label-setting algorithm based on label (3.17) is reduced to $O(\Gamma|A|B^{\Gamma+1})$ since these labels contain $\Gamma+1$ resources and their extension through new arcs can be done in $O(\Gamma)$ time with the help of formula (3.19). Finally, the next result states the new robust dominance rule.

**Lemma 4.** *Consider the $\mathcal{U}^\Gamma$-TWSP and let $z = (\kappa, \tau^0, \dots, \tau^\Gamma)$ and $z' = (\kappa', \tau'^0, \dots, \tau'^\Gamma)$ be two labels associated to paths $p$ and $p'$ ending at the same node. Assume the following conditions hold:*

1. *$\kappa \leq \kappa'$*
2. *$\tau^j \leq \tau'^j$, for each $j = 0,\dots,\Gamma$*
3. *and at least one inequality is strict.*

*Then, label $z'$ is dominated by label $z$.*

## 3.6 Computational experiments

We report below succinct numerical results on the grid networks from Class 6 used in Dumitrescu and Boland [2003]. The characteristics of these networks are reported in Table 3.1. For each network, we generate a set of different instances by varying the values of $\Gamma$ (or $\gamma$) and $\rho = \hat{u}_a/\overline{u}_a$ as follows. Ratio $\rho$ takes each value in $\{0.5, 0.6, 0.7, 0.8, 0.9, 1\}$. We compute $\Gamma$ and $\gamma$ for each probability level $\epsilon$ in $\{0.01, 0.05, 0.1\}$ following the construction outlined in Chapter 2.

| test | nodes | arcs | density |
|---|---|---|---|
| G1 | 625 | 2400 | 3.84 |
| G2 | 2500 | 9800 | 3.92 |
| G3 | 5625 | 22200 | 3.95 |
| G4 | 15625 | 62000 | 3.97 |
| G5 | 22500 | 89400 | 3.97 |
| G6 | 30625 | 121800 | 3.98 |
| G7 | 40000 | 159200 | 3.98 |

Table 3.1: Characteristics of the test problems.

**Generating time windows** We describe next how we generate $\underline{b}$ and $\overline{b}$ such that all instances always have a feasible solution. We let $\hat{T}$ and $\bar{T}$ represent the shortest path trees with respect to $\overline{u} + \hat{u}$ and $\overline{u}$, respectively. We let $\hat{p}$ denote the path from node $o$ to node $d$ in $\hat{T}$ and the width of time windows is denoted by $w$ (and set to a value in $\{40, 100, 200\}$). The values of $\underline{b}_i$ and $\overline{b}_i$ are set according to the following rule

$$\begin{cases} \underline{b}_i = \hat{T}_i - w; \overline{b}_i = \hat{T}_i, & \text{for each } i \in \hat{p}, \\ \underline{b}_i = \hat{T}_i - w; \overline{b}_i = \hat{T}_i, & \text{for each } i \notin \hat{p} \text{ and } rand < 0.5, \\ \underline{b}_i = \bar{T}_i; \overline{b}_i = \bar{T}_i + w, & \text{for each } i \notin \hat{p} \text{ and } rand \geq 0.5, \end{cases} \tag{3.20}$$

where $\hat{T}_i$ and $\bar{T}_i$ are the costs of the paths from $o$ to $i$ in $\hat{T}$ and $\bar{T}$, respectively, and *rand* is chosen randomly through the uniform distribution $[0, 1]$.

**Benefit of robustness** In our results, we report the optimal solution cost for the deterministic problems, while the solution costs for robust problems are provided under the benefit of robustness. Namely, in the absence of advanced framework to handle uncertainty, such as robust optimization or stochastic programming, decision makes tend to overestimate the parameters subject to uncertainty to avoid infeasible solutions, whose repairing cost would be very high in practice. Hence, they would solve a pessimistic deterministic problem where all uncertain parameters are set to their extreme values. The Benefit of Robustness can thus be computed as follows

$$\text{BoR} = \frac{\text{opt}(TWSP) - \text{opt}(P)}{\text{opt}(TWSP)},$$

where $\text{opt}(P)$ is the optimal solution cost of problem $P \in \{\mathcal{U}^{\Gamma}\text{-}TWSP, \mathcal{U}^{\gamma}\text{-}TWSP\}$.

**Results** Table 3.2 shows that the robust $TWSP$ with variable budgeted uncertainty is more difficult to solve than $\mathcal{U}^{\Gamma}$-$TWSP$. Indeed, the computational cost of the label-setting algorithm for solving the $\mathcal{U}^{\gamma}$-$TWSP$ is, on average, 1.52 times higher than the time required for solving the $\mathcal{U}^{\Gamma}$-$TWSP$. This behavior is justified by the number of generated labels. Indeed, the number of labels generated when solving $\mathcal{U}^{\gamma}$-$TWSP$ is 1.54

times higher than that processed by the label-setting algorithm to solve $\mathcal{U}^{\Gamma}$-$TWSP$. The higher number of labels is due to the fact that the feasibility check is done for a subset of the element of the labels. It follows that infeasible labels for the $\mathcal{U}^{\Gamma}$-$TWSP$ are feasible for the $\mathcal{U}^{\gamma}$-$TWSP$. The table also shows that model $\mathcal{U}^{\gamma}$-$TWSP$ leads to a significant increase in the benefit of robustness obtained by model $\mathcal{U}^{\Gamma}$-$TWSP$.

| | | $TWSP$ | | | $\mathcal{U}^{\Gamma}$-$TWSP$ | | | $\mathcal{U}^{\gamma}$-$TWSP$ | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $\epsilon$ | graphs | time | #labels | cost | time | #labels | BoR | time | #labels | BoR |
| 0.01 | G1 | 0.0 | 83 | 4897 | 0.0 | 109 | 3.00 | 0.0 | 262 | 5.11 |
| | G2 | 0.0 | 151 | 10215 | 0.2 | 489 | 4.02 | 0.6 | 1323 | 5.65 |
| | G3 | 0.1 | 232 | 14482 | 1.2 | 1388 | 5.94 | 3.1 | 3108 | 7.88 |
| | G4 | 1.5 | 436 | 24740 | 7.8 | 3127 | 5.38 | 21.2 | 7358 | 7.49 |
| | G5 | 2.5 | 471 | 16592 | 14.6 | 4034 | 2.57 | 40.1 | 9552 | 3.37 |
| | G6 | 4.4 | 525 | 29080 | 51.0 | 9271 | 6.17 | 124.9 | 20667 | 9.17 |
| | G7 | 6.9 | 594 | 24892 | 39.2 | 4929 | 2.92 | 360.6 | 15382 | 5.02 |
| AVG | | 2.2 | 356 | 17842 | 16.3 | 3335 | 4.29 | 78.7 | 8236 | 6.24 |
| 0.05 | G1 | 0.0 | 83 | 4897 | 0.0 | 141 | 3.04 | 0.0 | 258 | 5.19 |
| | G2 | 0.0 | 151 | 10215 | 0.2 | 455 | 4.43 | 0.4 | 985 | 6.03 |
| | G3 | 0.1 | 232 | 14482 | 1.3 | 1219 | 7.06 | 2.0 | 2059 | 9.38 |
| | G4 | 1.5 | 436 | 24740 | 7.6 | 2751 | 5.89 | 14.3 | 5336 | 7.87 |
| | G5 | 2.5 | 471 | 16592 | 12.6 | 3093 | 2.80 | 23.8 | 5901 | 4.02 |
| | G6 | 4.4 | 525 | 29080 | 37.5 | 6325 | 7.22 | 70.9 | 12058 | 9.56 |
| | G7 | 6.9 | 594 | 24892 | 34.5 | 4316 | 3.84 | 80.0 | 10177 | 5.67 |
| AVG | | 2.2 | 356 | 17842 | 13.4 | 2614 | 4.90 | 27.4 | 5253 | 6.82 |
| 0.1 | G1 | 0.0 | 83 | 4897 | 0.0 | 145 | 3.98 | 0.0 | 226 | 5.19 |
| | G2 | 0.0 | 151 | 10215 | 0.2 | 441 | 4.90 | 0.4 | 900 | 6.47 |
| | G3 | 0.1 | 232 | 14482 | 1.0 | 1040 | 7.38 | 1.6 | 1661 | 10.32 |
| | G4 | 1.5 | 436 | 24740 | 6.6 | 2211 | 7.30 | 12.1 | 4039 | 8.48 |
| | G5 | 2.5 | 471 | 16592 | 10.6 | 2518 | 3.18 | 19.1 | 4417 | 4.23 |
| | G6 | 4.4 | 525 | 29080 | 29.0 | 4934 | 7.78 | 46.5 | 8108 | 10.02 |
| | G7 | 6.9 | 594 | 24892 | 28.5 | 3580 | 4.20 | 57.9 | 7779 | 6.15 |
| AVG | | 2.2 | 356 | 17842 | 10.8 | 2124 | 5.53 | 19.7 | 3876 | 7.27 |

Table 3.2: Average computational times, number of labels, and cost reductions.

# Chapter 4

# Scheduling

*The chapter is based on Bougeret et al. [2016].*

## 4.1 Introduction

Scheduling is a very wide topic in combinatorial optimization with applications ranging from production and manufacturing systems to transportation and logistics systems. Stated generally, the objective of scheduling is to allocate optimally scarce resources to activities over time. The practical relevance and the difficulty of solving the general scheduling problem have motivated an intense research activity in a large variety of scheduling environments. Scheduling problems are usually defined in the following way: given a set of $n$ jobs represented by $\mathcal{J}$, a set of $m$ machines represented by $\mathcal{M}$, and processing times represented by the tuple $u$, we look for a schedule $x$ of the jobs on the machines that satisfies the side constraints, represented by the set $\mathcal{X}^{comb}$ of feasible schedules, and minimize objective function $f(x, u)$. Formally, this amounts to solve optimization problem

$$\min_{x \in \mathcal{X}^{comb}} f(x, u).$$

As before, we suppose in this chapter that the processing times $u$ are uncertain and address optimization problems of the form

$$\min_{x \in \mathcal{X}^{comb}} \max_{u \in \mathcal{U}^{\Gamma}} f(x, u), \tag{4.1}$$

where the exact definitions of $\mathcal{X}^{comb}$ and $f$ depend on each scheduling problem. Differently from $\mathcal{U}^{\Gamma}$-$CO$, we see that scheduling problems represented abstractly in (4.1) feature nonlinear objective functions in general. This explains why Theorems 2 and 3 do not apply to these problems, opening the way for ad-hoc exact and approximation algorithms.

Recall the three-field notation $\alpha|\beta|\gamma$ from Graham et al. [1979] where $\alpha$ describes the machine environment, $\beta$ the job characteristics, and $\gamma$ the objective function. We focus

on the following classical scheduling problems in this chapter. Let $C_j(x,u)$ denote the completion time of task $j$ for schedule $x$ and processing times represented by $u$. The first type of problems studied herein concerns the minimization of the weighted sum of completion times on a single machine ($1||\sum_j w_jC_j$), defined by letting $\mathcal{X}^{comb}$ contain all orders for the $n$ tasks and setting $f(x,u) = \sum_{j\in\mathcal{J}} w_jC_j(x,u)$. We pay a particular attention to the case where $w_j = 1$ for each $j \in \mathcal{J}$, which is denoted $1||\sum_j C_j$. The second type of problems studied herein considers a set of $m$ machines, which can be identical ($P$), uniform ($Q$) or unrelated ($R$), and minimize the makespan $f(x,u) = C_{max}(x,u) = \max_{j\in\mathcal{J}} C_j(x,u)$. In the first case, processing job $j$ on machine $i$ is given by $u_j$. In the second case, the processing time is given by $u_{ij} = u_j/s_i$ where $s_i$ is the speed of machine $i$. In the last case, the processing times are given by an arbitrary matrix $u \in \mathbb{N}^{m\times n}$. The resulting problems are denoted by $P||C_{max}$, $Q||C_{max}$, and $R||C_{max}$, respectively. In these problems, $\mathcal{X}^{comb}$ contain all assignments of the $n$ tasks to the $m$ machines.

We detail below our contributions more specifically as well as the structure of the chapter. Let us extend Graham's notation to $\alpha|\beta|\mathcal{U}_p^\Gamma|\gamma$ to specify that the cost of any feasible schedule is obtained for the worst processing times in $\mathcal{U}^\Gamma$. In Section 4.2 we consider one machine problems minimizing the sum of completion times. We prove that $1||\mathcal{U}_p^\Gamma|\sum C_j$ is polynomial by extending Theorem 2. Comparing with Aloulou and Croce [2008], Daniels and Kouvelis [1995], Yang and Yu [2002], the result illustrates how $\mathcal{U}^\Gamma$-robust scheduling can lead to more tractable problems than robust scheduling with arbitrary uncertainty sets. We show then that $1||\mathcal{U}_p^\Gamma|\sum w_jC_j$ is weakly $\mathcal{NP}$-hard if $\Gamma = 1$ and strongly $\mathcal{NP}$-hard if $\Gamma > 1$. To our knowledge, this is the first example of a polynomial scheduling problem having a $\mathcal{NP}$-hard $\mathcal{U}^\Gamma$-robust counterpart. In Section 4.3 we show that $P||\mathcal{U}_p^\Gamma|C_{max}$ is 3-approximable. Section 4.4 is dedicated to $R||\mathcal{U}_p^\Gamma|C_{max}$. We provide an average $O(\log m)$-approximation based on an extended formulation of the problem. The formulation is solved in polynomial time by combining column generation with an approximately feasible solution for the pricing problem. Finally, a classical randomized rounding is applied, which is carefully analyzed to provide the required approximation factor.

## 4.2 Minimizing sum of completion times

### 4.2.1 Unitary weights

Problem $1||\sum C_j$ is one of the simplest scheduling problem, yet its robust version is $\mathcal{NP}$-hard in the weak sense for arbitrary uncertainty sets $\mathcal{U}$, even for two scenarios Yang and Yu [2002]. In contrast, we show below that the $\mathcal{U}^\Gamma$-robust version of the problem can be solved in polynomial time.

Our approach applies an extension of Theorem 2 to problem $1||\mathcal{U}_p^\Gamma|\sum C_j$. Let $x_{ij}$ be

equal to 1 iff job $j$ is scheduled in position $i$. Problem $1||\mathcal{U}_p^\Gamma| \sum C_j$ can be cast as

$$\min \left\{ \max_{u \in \mathcal{U}^\Gamma} \sum_{(i,j) \in \mathcal{J} \times \mathcal{J}} u_j (n+1-i) x_{ij} : \sum_{i \in \mathcal{J}} x_{ij} = 1, j \in \mathcal{J}, \sum_{j \in \mathcal{J}} x_{ij} = 1, i \in \mathcal{J} \right\}. \quad (4.2)$$

**Observation 1.** *Theorem 2 cannot be applied to problem* (4.2) *because its cost function is defined by a product of parameters $u_j q_i$ where only $u \in \mathcal{U}^\Gamma$.*

We now provide an extension of Theorem 2, which encompasses problem (4.2).

**Theorem 11.** *Let $\mathcal{X} \subseteq \left\{ \{0,1\}^{I \times J} : \sum_{i=1}^{I} x_{ij} = 1,\ j = 1, \ldots, J \right\}$ and let $q \in \mathbb{R}^I$ and $u \in \mathcal{U}^\Gamma$ be cost vectors. The optimal solution to problem*

$$\min_{x \in \mathcal{X}} \max_{u \in \mathcal{U}^\Gamma} \sum_{i,j} u_j q_i x_{ij} \quad (4.3)$$

*can be obtained by solving the problems $\min_{x \in \mathcal{X}} \sum_{i,j} (\overline{u}_j + \hat{u}_j) q_i x_{ij}$ and $\min_{x \in \mathcal{X}} \sum_{i,j} \left( \overline{u}_j + \breve{u}_{ij}^{kl} \right) q_i x_{ij}$, for each $k \in I, l \in J$, where $\breve{u}_{ij}^{kl} = \max(0, \hat{u}_j - \frac{\hat{u}_l q_k}{q_i})$.*

*Proof.* The proof follows closely the lines of the proof of Theorem 2 from Bertsimas and Sim [2003]. Let us detail the inner maximization of (4.3) as

$$\sum_{i,j} \overline{u}_j q_i x_{ij} \quad + \quad \max \Big\{ \quad \sum_{i,j} \delta_j \hat{u}_j q_i x_{ij} \ : \\ \sum_j \delta_j \leq \Gamma, \\ \delta_j \in \{0,1\}, \quad j = 1, \ldots, J \quad \Big\}.$$

Removing the binary conditions on $\delta$ in the definition of $\mathcal{U}^\Gamma$, one obtains a polytope whose extreme points correspond with the elements of $\mathcal{U}^\Gamma$. Hence if we consider the linear programming relaxation of the above problem, which is equal to the solution cost of its dual

$$\min \Big\{ \quad \Gamma\theta + \sum_j y_j \ : \\ \theta + y_j \geq \sum_i \hat{u}_j q_i x_{ij}, \quad j = 1, \ldots, J \\ \theta, y \geq 0 \quad \Big\}.$$

Substituting $y_j$ by $\max(0, \sum_i \hat{u}_j q_i x_{ij} - \theta)$, we can further reformulate (4.3) as

$$\min_{x \in \mathcal{X}, \theta \geq 0} \Gamma\theta + \sum_{i,j} \overline{u}_j q_i x_{ij} + \sum_j \max(0, \sum_i \hat{u}_j q_i x_{ij} - \theta). \quad (4.4)$$

The crucial step of our proof (which differs from Theorem 2) is that, because the constraint $\sum_{i=1}^{I} x_{ij} = 1$ holds for each $j = 1, \ldots, J$, we can further reformulate (4.4) as

$$\min_{x \in \mathcal{X}, \theta \geq 0} \Gamma\theta + \sum_{i,j} \overline{u}_j q_i x_{ij} + \sum_{j} \sum_{i} x_{ij} \max(0, \hat{u}_j q_i - \theta).$$

Introducing $r = (i, j)$ and renaming variable $x_{ij}$ as $z_r$, products $\overline{u}_j q_i$ and $\hat{u}_j q_i$ as $c_r$ and $d_r$, respectively, the rest of the proof is identical to the proof of Theorem 2 from Bertsimas and Sim [2003]. □

Applying Theorem 11 to (4.2), we obtain that $1||\mathcal{U}_p^\Gamma|\sum C_j$ can be solved by solving $O(n^2)$ assignment problems. We point out that, although the robust problem can be solved in polynomial time, the modified cost coefficients $\overline{u}_j + \breve{u}_{ij}^{kl}$ break the structure of $1||\mathcal{U}_p^\Gamma|\sum C_j$, i.e., the deterministic problems with cost vector $\overline{u} + \breve{u}^{kl}$ are not instances of $1||\sum C_j$.

### 4.2.2 General weights

It is well known that problem $1||\sum w_j C_j$ can be solved in polynomial time by applying Smith's rule [Smith, 1956] (i.e., scheduling jobs by non-decreasing $\frac{u_j}{w_j}$). However, it does not seem easy to extend that simple rule to the robust problem $1||\mathcal{U}_p^\Gamma|\sum w_j C_j$. In fact, we show that the problem is $\mathcal{NP}$-hard in the weak sense for $\Gamma = 1$ and strongly $\mathcal{NP}$-hard for arbitrary $\Gamma$. For that, we need the following two lemmas.

**Lemma 5.** *Given $X \subset \mathcal{J}$ such that $\frac{w_j}{\overline{u}_j} < \frac{w_\ell}{\overline{u}_\ell + \hat{u}_\ell}$, $\forall j \in X$, and $\ell \in \mathcal{J} \setminus X$, in any optimal solution for $1||\mathcal{U}_p^\Gamma|\sum w_j C_j$ the jobs in $X$ are the last $|X|$ in the schedule.*

*Proof.* We prove the proposition by contradiction. Assume that there is an optimal solution $\sigma^*$ for $1||\mathcal{U}_p^\Gamma|\sum w_j C_j$ where two consecutive jobs $j$ and $\ell$ have

$$\frac{w_j}{\overline{u}_j} < \frac{w_\ell}{\overline{u}_\ell + \hat{u}_\ell}. \tag{4.5}$$

Let $c^*(u)$ denote the solution cost for the specific vector $u \in \mathcal{U}^\Gamma$ and $c^*$ be the solution cost for worst deviations; that is, $c^* = \max_{u \in \mathcal{U}^\Gamma} c^*(u)$. By swapping $j$ and $\ell$ in $\sigma^*$, we obtain an alternative schedule $\sigma'$ with cost denoted $c'$, which satisfies

$$\begin{aligned} c' &= \max_{u \in \mathcal{U}^\Gamma}(c^*(u) + u_\ell w_j - u_j w_\ell) \\ &\leq \max_{u \in \mathcal{U}^\Gamma} c^*(u) + \max_{u \in \mathcal{U}^\Gamma}(u_\ell w_j - u_j w_\ell) \\ &\leq c^* + (\overline{u}_\ell + \hat{u}_\ell) w_j - \overline{u}_j w_\ell. \end{aligned} \tag{4.6}$$

From (4.5) and (4.6), we obtain that $c' < c^*$, which contradicts the optimality of $\sigma^*$. □

**Lemma 6.** *There exists an optimal solution for* $1||\mathcal{U}_p^\Gamma|\sum w_jC_j$ *where, for any two jobs* $j$ *and* $\ell$ *with* $\overline{u}_j = \overline{u}_\ell$, $w_j = w_\ell$, *and* $\hat{u}_j < \hat{u}_\ell$, $j$ *is scheduled before* $\ell$.

*Proof.* Let $j$ and $\ell$ be two jobs satisfying the conditions of this proposition such that $\ell$ precedes $j$ in an optimal solution $\sigma^*$. We show that swapping $\ell$ and $j$ does not increase the robust cost $c^*$ of $\sigma^*$. Let $\sigma'$ be the resulting schedule. Let also $W_k$ be the sum of weights of all jobs that do not precede $k$ in $\sigma^*$, for all $k \in \mathcal{J}$. Clearly $W_j < W_\ell$. Moreover, the total cost due to mean processing times is the same for $\sigma^*$ and $\sigma'$, and the total cost due to deviations is calculated by selecting the $\Gamma$ jobs with maximum $\hat{u}_kW_k$ among all $k \in \mathcal{J}$, and summing up these values. After the swap, $\hat{u}_jW_j$ and $\hat{u}_\ell W_\ell$ are replaced by $\hat{u}_\ell W_j$ and $\hat{u}_jW_\ell$, and the remaining values are kept unchanged. Since $W_j < W_\ell$ and $\hat{u}_j < \hat{u}_\ell$, we have that $\hat{u}_\ell W_j + \hat{u}_jW_\ell < \hat{u}_jW_j + \hat{u}_\ell W_\ell$. As a result, the total cost due to deviations cannot increase after the swap. □

For the hardness proof, we define the $k$-PARTITION problem.

**Definition 1.** *Given* $kN$ *positive numbers* $a_1, \ldots, a_{kN}$ *satisfying* $\sum_{j=1}^{kN} a_j = NA$, *k-PARTITION asks if there exists a partition of* $\{a_1, \ldots, a_{kN}\}$ *into* $N$ *subsets* $S_1, \ldots, S_N$ *such that* $\sum_{j\in S_i} a_j = A$, *for* $i = 1, \ldots, N$.

The decision version of $1||\mathcal{U}_p^\Gamma|\sum w_jC_j$, denoted by $(1||\mathcal{U}_p^\Gamma|\sum w_jC_j, K)_{dec}$, asks for a schedule whose robust cost is not greater than a given integer $K$.

**Theorem 12.** *There is a polynomial reduction from k-PARTITION to* $(1||\mathcal{U}_p^\Gamma|\sum w_jC_j, K)_{dec}$ *with* $\Gamma = N - 1$.

*Proof.* First, we describe a reduction allowing that $\hat{u}$ is a vector of rational numbers. Later, we show how the proposed reduction can be modified to use only integer numbers, and still satisfy the conditions of this theorem. We create three types of jobs. For $j = 1, \ldots, kN$, job $j$, referred to as a *partition* job, has $w_j = \overline{u}_j = a_j$, and $\hat{u}_j = 0$; for $j = kN + 1, \ldots, (k + 1)N - 1$, job $j$, referred to as a *tail* job, has $w_j = 1$, $\overline{u}_j = 2N$, and $\hat{u}_j = \frac{4NA}{(k+1)N-j}$; for $j = (k + 1)N, \ldots, (k + 2)N - 2 = n$, job $j$, referred to as a *separating* job, has $w_j = 2$, $\overline{u}_j = 1$, and $\hat{u}_j = \frac{4NA}{W_j+\beta(j)A}$, where $\beta(j) = j - (k + 1)N + 1$, and $W_j = N - 1 + 2\beta(j)$. Moreover, $\Gamma = N - 1$.

We restrict our analysis to schedules that satisfy Lemmas 5 and 6 since they necessarily include an optimal solution to the optimization version of the problem. Thus, we can conclude that the last $N - 1$ scheduled jobs are exactly the tail jobs, which are sorted in an increasing order by their indices, and that the separating jobs are sorted in a decreasing order by their indices.

For a given schedule $\sigma$, let $\sigma(\ell)$ denote the $\ell$-th job to be executed, for $\ell = 1, \ldots, kN$, and define $\sigma^{-1}(j)$ such that $\sigma(\sigma^{-1}(j)) = j$ for each $j \in \mathcal{J}$. Define also $u^\sigma$ as the worst vector $u \in \mathcal{U}^\Gamma$ for the schedule $\sigma$. In the objective function $\sum_{j\in\mathcal{J}} \sum_{\ell=\sigma^{-1}(j)}^{kN} u_j^\sigma w_{\sigma(\ell)}$, the term $u_j^\sigma w_{\sigma(\ell)}$ is referred to as the cost from job $j$ to job $\sigma(\ell)$. Let also $\sigma^\Delta(\ell)$ denote the

$\ell$-th partition job to be executed according to $\sigma$, and define $(\sigma^{\Delta})^{-1}$ analogously to $\sigma^{-1}$. Finally, let

$$A_j^{\sigma} = \sum_{\substack{\ell=1 \\ \sigma^{-1}(\ell) \geq j}}^{kN} a_\ell$$

be the sum of the weights of the partitions jobs scheduled after job $j$ (and including the weight of job $j$ if it is a partition job).

We divide the cost of a schedule $\sigma$ for the created instance of $(1||\mathcal{U}_p^{\Gamma}| \sum w_j C_j, K)_{dec}$ into six terms:

- the cost from partition jobs to partition jobs, given by

$$c_1 = \sum_{j=1}^{kN} \sum_{\ell=(\sigma^{\Delta})^{-1}(j)}^{kN} \overline{u}_j w_{\sigma^{\Delta}(\ell)} = \sum_{j=1}^{kN} \sum_{\ell=1}^{kN} a_j a_\ell;$$

- the cost from partition jobs to tail jobs, given by

$$c_2 = \sum_{j=1}^{kN} \sum_{\ell=kN+1}^{(k+1)N-1} \overline{u}_j w_\ell = NA(N-1);$$

- the cost from tail jobs excluding deviations, given by

$$c_3 = \sum_{j=kN+1}^{(k+1)N-1} \sum_{\ell=j}^{(k+1)N-1} \overline{u}_j w_{\sigma(\ell)} = N^2(N-1);$$

- the cost from partition jobs to separating jobs, given by

$$c_4 = \sum_{j=1}^{kN} \sum_{\substack{\ell=(k+1)N \\ \sigma^{-1}(\ell) > \sigma^{-1}(j)}}^{(k+2)N-2} \overline{u}_j w_\ell = 2NA(N-1) - 2 \sum_{j=(k+1)N}^{(k+2)N-2} A_j^{\sigma};$$

- the cost from separating jobs excluding deviations, given by

$$c_5 = \sum_{j=(k+1)N}^{(k+2)N-2} \sum_{\substack{\ell=1 \\ \sigma^{-1}(\ell) > \sigma^{-1}(j)}}^{(k+2)N-2} \overline{u}_j w_\ell = N(N-1) + (N-1)^2 + \sum_{j=(k+1)N}^{(k+2)N-2} A_j^{\sigma};$$

- the cost due to deviations from both the separating jobs and the tail jobs, given by

$$c_6 = \sum_{j=kN+1}^{(k+2)N-2} \sum_{\substack{\ell=1 \\ \sigma^{-1}(\ell) > \sigma^{-1}(j)}}^{(k+2)N-2} (u_j^{\sigma} - \overline{u}_j) w_\ell$$

$$= \sum_{j=(k+1)N}^{(k+2)N-2} \max \left\{ 4NA, \sum_{\substack{\ell=1 \\ \sigma^{-1}(\ell)>\sigma^{-1}(j)}}^{(k+2)N-2} \hat{u}_j w_\ell \right\}$$

$$= \sum_{j=(k+1)N}^{(k+2)N-2} \max \left\{ 4NA, \frac{4NA}{W_j + \beta(j)A}(W_j + A_j^\sigma) \right\}$$

where the second equality holds because for each tail job, the cost due to its deviation is equal to $4NA$.

The total cost is given by $c_1 + c_2 + c_3 + c_4 + c_5 + c_6$. Note that only $c_6$, the third term of $c_5$ and the second term of $c_4$ depend on the schedule $\sigma$. All remaining terms are constant. Summing up the non-constant terms, we obtain

$$\tilde{c}(\sigma) = \sum_{j=(k+1)N}^{(k+2)N-2} \max \left\{ 4NA - A_j^\sigma, \frac{4NA}{W_j + \beta(j)A} W_j + \left( \frac{4NA}{W_j + \beta(j)A} - 1 \right) A_j^\sigma \right\}.$$

Assuming that $A > 3$, we have that $\frac{4NA}{W_j+\beta(j)A} > 2$. Hence, the value of $\tilde{c}(\sigma)$ is minimized (and thus the total cost) when $\beta(j)A = A_j^\sigma$, for $j = (k+1)N, \ldots, (k+2)N-2$. This only occurs when each sum of processing times of partition jobs scheduled between two consecutive separating jobs is exactly $A$. Otherwise, by the coefficients to $A_j^\sigma$ in the two arguments of the maximum function, $\tilde{c}(\sigma)$ increases by at least one unit. Thus, setting $K = c_1 + c_2 + c_3 + N(N-1) + (N-1)^2 + 5.5NA(N-1) + 0.5$, we have that a positive answer to $k$-PARTITION yields a schedule of cost $K - 0.5$, and that any schedule costs at least $K + 0.5$ otherwise.

To ensure that the constructed instance contains only integer numbers on the input, we multiply all processing times and $K$ by $2(N-1)\sum_{j\in\mathcal{J}} w_j$. This yields a solution of cost $K - (N-1)\sum_{j\in\mathcal{J}} w_j$ in the case of a positive answer to $k$-PARTITION, and no solution of cost less than $K + (N-1)\sum_{j\in\mathcal{J}} w_j$ otherwise. By rounding up the values of $\hat{u}_j$, for $j = kN+1, \ldots, (k+2)N-2$, the cost of each solution may increase by at most $(N-1)\sum_{j\in\mathcal{J}} w_j$, still allowing to answer $k$-PARTITION. Moreover, if $A$ is polynomially bounded for the $k$-PARTITION, so are all input data for the constructed instance. □

The next corollary proves the desired hardness results.

**Corollary 4.** *$(1||\mathcal{U}_p^\Gamma|\sum w_j C_j, K)_{dec}$ is $\mathcal{NP}$-complete in the weak sense for $\Gamma = 1$ and strongly $\mathcal{NP}$-complete when $\Gamma$ is part of the input.*

*Proof.* For $N = 2$ and arbitrary $k$, $k$-PARTITION corresponds to PARTITION, which is weakly $\mathcal{NP}$-complete, and, for $k = 3$ and arbitrary $N$, $k$-PARTITION generalizes 3-PARTITION, which is $\mathcal{NP}$-complete in the strong sense. Hence, the corollary follows directly from the reduction given by Theorem 12. □

## 4.3 Minimizing makespan on identical machines

We introduce the following notations. A schedule is denoted by $\sigma$ and $\sigma_i \subseteq \mathcal{J}$ denotes a schedule restricted to machine $i$. An optimal schedule is denoted by $\sigma^*$ and its value is denoted by opt. For any set of jobs $X \subseteq \mathcal{J}$, we use $\overline{u}(X) = \sum_{j \in X} \overline{u}_j$ and $\hat{u}(X) = \sum_{j \in X} \hat{u}_j$. We let $\Gamma(X)$ contain the $\Gamma$ jobs from $X$ with highest deviations, $\hat{u}^\Gamma(X) = \hat{u}(\Gamma(X))$, and $\hat{u}^\Gamma(\sigma) = \sum_{i \in \mathcal{M}} \hat{u}^\Gamma(\sigma_i)$. We also use $C(\mathcal{J}) = \overline{u}(\mathcal{J}) + \hat{u}^\Gamma(\mathcal{J})$ and $C(\sigma) = \max_{i \in \mathcal{M}} C(\sigma_i)$.

We say that an algorithm $A$ is a $\rho$-dual approximation if for any $\omega$ and instance $\mathcal{I}$, either $A(\mathcal{I}, \omega)$ builds a schedule $\sigma$ such that $C(\sigma) \leq \rho\omega$ or fails, which implies then that $\omega < \text{opt}$. Notice that any $\rho$-dual approximation can be converted to a $\rho$-approximation algorithm by performing a binary search on $\omega$ to find the smallest $\omega$ that is not rejected.

Let us now design an algorithm $A$ for $P||\mathcal{U}_p^\Gamma|C_{max}$, and prove the following theorem.

**Theorem 13.** *Algorithm $A$ is a 3 dual approximation algorithm for $P||\mathcal{U}_p^\Gamma|C_{max}$. This implies that $P||\mathcal{U}_p^\Gamma|C_{max}$ admits a 3-approximation in the general case.*

Before presenting algorithm $A$, we point out an important obstacle faced when designing dual algorithms for the problem. As usual, fixing the value of $\omega$ is suitable as it defines the size of bins in which we can schedule the jobs. Thus, a natural way to design a dual approximation algorithm would be to take the jobs in an arbitrary order and schedule as many of them as possible into each machine, moving to the next machine whenever $C(\sigma_i) > \omega$, and rejecting $\omega$ if there remain some jobs after filling $m$ machines. This algorithm would not exceed $2\omega$, thus improving over Theorem 13. However, this algorithm is not correct, as the existence of a $\sigma$ with $C(\sigma_i) > \omega$ for any $i$ does not imply that $\omega < \text{opt}$, even if the algorithm selects jobs by non-increasing $\hat{u}_j$. Indeed, consider the input where $m = \Gamma = 2$, $\omega = 15$ and $u_1 = (0, 10)$, $u_2 = u_3 = u_4 = (0, 6)$, $u_5 = (5, 0)$, $u_6 = (3, 0)$ (where $u_j = (\overline{u}_j, \hat{u}_j)$). The previous algorithm would create $\sigma_1 = \{1, 2\}$, $\sigma_2 = \{3, 4, 5\}$ and rejects as $C(\sigma_i) > \omega$ for any $i$ and not all jobs are scheduled, whereas there exists a schedule $\sigma_1^* = \{1, 5\}$, $\sigma_2^* = \{2, 3, 4, 6\}$ that fits in $\omega$. This explains the design of Algorithm 1. The validity of the algorithm is shown in the rest of this section.

**Observation 2.** *For any $\sigma$, $C(\sigma) \leq \omega \Rightarrow \hat{u}^\Gamma(\sigma) + \overline{u}(\mathcal{J}) \leq m\omega$.*

**Lemma 7.** *For any $i$, $C(\sigma_i) \leq 3\omega$*

*Proof.* In the worst case, before adding the last job $j$ in the interior while loop we had $\overline{u}(X) = \omega$ and $\hat{u}^\Gamma(X) = \omega$, and thus $C(\sigma_i) \leq 2\omega + \overline{u}_j + \hat{u}_j$ with $\overline{u}_j + \hat{u}_j \leq \omega$ (if there is a job with $\overline{u}_j + \hat{u}_j > \omega$, we can immediately reject $\omega$). □

**Lemma 8.** *If $A$ fails, then* $\text{opt} > \omega$.

*Proof.* Let us suppose that $A$ fails and suppose by contradiction that $\text{opt} \leq \omega$. We say that machine $i$ is of type 1 iff $\hat{u}^\Gamma(\sigma_i) > \omega$, and is of type 2 otherwise. Notice that a schedule

**Algorithm 1** Algorithm $A$

```
// Given a set of jobs J, A(J) either schedules J on m machines, or fails.
i = 1
while J ≠ ∅ AND i ≤ m do
  σ_i ← ∅
  while J ≠ ∅ AND ū(σ_i) ≤ ω AND û^Γ(σ_i) ≤ ω do
    assign to σ_i the largest job (in term of û_j) of J;
  end while
  i ← i + 1;
end while
if J ≠ ∅ then
  fails
end if
```

on a machine of type 1 contains at most $\Gamma$ jobs (as jobs are added by non-increasing $\hat{u}_j$), and a schedule $\sigma_i$ on a machine of type 2 verifies $\overline{u}(\sigma_i) > w$. Let $\mathcal{M}_1$ be the set of machines of type 1, and let $\mathcal{J}_1$ be the set of jobs scheduled by $A$ in machines $\mathcal{M}_1$. Let $\mathcal{M}_2$ and $\mathcal{J}_2$ be defined in the same way. We have

- $\overline{u}(\mathcal{J}_2) > |\mathcal{M}_2|\omega$ by definition of type 2
- $\hat{u}(\mathcal{J}_1) > |\mathcal{M}_1|\omega$ by definition of type 1
- $\hat{u}(\sigma^*) \geq \hat{u}(\mathcal{J}_1)$

Let us prove the last item. Notice first that for any schedule $\sigma'$ of $\mathcal{J}_1$ on $m$ machines such that $C(\sigma') \leq \omega$, $\hat{u}(\sigma') = \hat{u}(\mathcal{J}_1)$. Indeed, let $i$ be the last machine in $\mathcal{M}_1$ and let $x = |\sigma_i|$. Notice that as we select the jobs by non-increasing order of $\hat{u}_j$ in the interior while loop, $\sigma_i$ contains the $x$ smallest jobs (in terms of $\hat{u}_j$) of $\mathcal{J}_1$. As $i$ is type 1 we get $\hat{u}^\Gamma(\sigma_i) > \omega$, and we deduce that in any schedule of $\mathcal{J}_1$ that fits in $\omega$, there are at most $x \leq \Gamma$ jobs on every machine. Thus, all jobs deviate in $\sigma'$, and $\hat{u}(\sigma') = \hat{u}(\mathcal{J}_1)$.

Then, notice that $\hat{u}(\sigma^*) \geq \hat{u}(\sigma^*_{|\mathcal{J}_1})$ where $\sigma^*_{|\mathcal{J}_1}$ is the schedule we obtain by starting from $\sigma^*$ and only keeping jobs of $\mathcal{J}_1$ on each machine (and removing idle time). Thus, as $\text{opt}_{|\mathcal{J}_1}$ is a schedule of $\mathcal{J}_1$ on $m$ machines that fits in $\omega$, we know that $\hat{u}(\sigma^*_{|\mathcal{J}_1}) = \hat{u}(\mathcal{J}_1)$, concluding the proof of the last item.

Thus, we get $\overline{u}(\mathcal{J}) + \hat{u}(\sigma^*) \geq \overline{u}(\mathcal{J}_2) + \hat{u}(\mathcal{J}_1) > m\omega$, and thus according to Observation 2 we deduce $\text{opt} > \omega$, a contradiction. □

Lemmas 7 and 8 directly imply Theorem 13.

## 4.4 Minimizing makespan on unrelated machines

In this section, we denote by $\bar{u}_{ij}$ and $\hat{u}_{ij}$ respectively the mean and deviating processing times for job $j \in \mathcal{J} = \{1, \ldots, n\}$ on machine $i \in \mathcal{M} = \{1, \ldots, m\}$.

We provide an algorithm that yields an average $O(\log m)$ approximation factor. Notice that the straightforward generalization of the formulation from Lenstra et al. [1990] is not useful in the robust context because its fractional solution may contain up to $nm$ fractional variables. Hence, we must use a different approach, based on the extended formulation described next.

Define, for each $i \in \mathcal{M}$ and each $\nu \subseteq \mathcal{J}$, $\lambda_{i\nu} = 1$ if the set of jobs executed on machine $i$ is precisely $\nu$, and zero otherwise. Let $\mu(j, \nu) = 1$ if $j \in \nu$, and zero otherwise, and $\alpha(i, \nu) = \max\left\{\sum_{j\in\nu}(\bar{u}_{ij} + \xi_j\hat{u}_{ij}) : \xi \in \{0,1\}^n, \sum_{j\in\mathcal{J}} \xi_j \leq \Gamma\right\}$. The formulation follows:

$$
\min\Bigg\{ \quad \omega \ : \\
\sum_{i\in\mathcal{M}}\sum_{\nu\subseteq\mathcal{J}} \mu(j,\nu)\lambda_{i\nu} = 1, \qquad \forall j \in \mathcal{J} \\
\omega \geq \sum_{\nu\subseteq\mathcal{J}} \alpha(i,\nu)\lambda_{i\nu}, \qquad \forall i \in \mathcal{M} \\
\sum_{\nu\subseteq\mathcal{J}} \lambda_{i\nu} = 1, \qquad \forall i \in \mathcal{M} \\
\lambda_{i\nu} \in \{0,1\}, \qquad \forall (i,\nu) \in \mathcal{M} \times 2^{\mathcal{J}} \quad \Bigg\}.
$$

As with the formulation from Lenstra et al. [1990], the value of the lower bound improves if we drop the objective function and remove all variables $\lambda_{i\nu}$ such that $\alpha(i, \nu)$ is greater than a given target makespan value $\omega$. Namely, we consider the lower bound for $R||\mathcal{U}^{\Gamma}|C_{max}$ defined as follows

$$
LB = \{\min \omega : FP(\omega) \text{ is feasible}\}, \tag{4.7}
$$

where $FP(\omega)$ is defined by the following linear constraints:

$$
\sum_{i\in\mathcal{M}} \sum_{\substack{\nu\subseteq\mathcal{J} \\ \alpha(i,\nu)\leq\omega}} \mu(j,\nu)\lambda_{i\nu} = 1, \quad \forall j \in \mathcal{J} \tag{4.8}
$$

$$
\sum_{\substack{\nu\subseteq\mathcal{J} \\ \alpha(i,\nu)\leq\omega}} \lambda_{i\nu} = 1, \quad \forall i \in \mathcal{M} \tag{4.9}
$$

$$
\lambda_{i\nu} \geq 0, \quad \forall (i,\nu) \in \mathcal{M} \times 2^{\mathcal{J}}. \tag{4.10}
$$

$LB$ is a lower bound because the integrality restrictions on $\lambda$ have been relaxed. We show below how we can assert in polynomial time whether $FP(\omega)$ is infeasible or prove its feasibility for $2\omega$. This algorithm can be further combined with a binary search on the minimum value $\omega$ for which $FP(\omega)$ is feasible, yielding the following result.

**Theorem 14.** *We can compute in polynomial time a 2-approximate solution for LB.*

*Proof.* We solve problem (4.7) using a dual-approximation algorithm. Namely, for each value of $\omega$, either we show that $FP(2w)$ is feasible or that $FP(\omega)$ is infeasible. Then, the minimum value of $\omega$ for which $FP(\omega)$ is feasible that leads to zero objective value can be found through a binary search.

Let $\omega$ be the current value of the guess. We can check the feasibility of $FP(2\omega)$ by adding artificial variables $s_j$ that allow penalized infeasibilities, leading to the following linear program.

$$\min \Bigg\{ \quad \sum_{j \in \mathcal{J}} s_j \ : \tag{4.11}$$

$$\sum_{i \in \mathcal{M}} \sum_{\substack{\nu \subseteq \mathcal{J} \\ \alpha(i,\nu) \leq \omega}} \mu(j,\nu)\lambda_{i\nu} + s_j = 1, \qquad \forall j \in \mathcal{J} \tag{4.12}$$

$$\sum_{\substack{\nu \subseteq \mathcal{J} \\ \alpha(i,\nu) \leq \omega}} \lambda_{i\nu} = 1, \qquad \forall i \in \mathcal{M} \tag{4.13}$$

$$\lambda_{i\nu} \geq 0, \qquad \forall (i,\nu) \in \mathcal{M} \times 2^{\mathcal{J}}, \alpha(i,\nu) \leq \omega \tag{4.14}$$

$$s_j \geq 0, \qquad \forall j \in \mathcal{J} \Bigg\} \tag{4.15}$$

The continuous relaxation of the previous formulation can be solved in polynomial time, using for instance the Ellipsoid method [Khachiyan, 1980], if the problem of pricing the $\lambda$ variables is also polynomially solvable [Grötschel et al., 1988]. Such a pricing problem can be stated as follows. Let $\pi_j$, and $\theta_i$ be dual variables associated to constraints (4.12), and (4.13), respectively. The reduced cost of the variable $\lambda_{i\nu}$, denoted by $\bar{c}(\lambda_{i\nu})$, is equal to $-\sum_{j \in \nu} \pi_j - \theta_i$.

Then, for each $i \in \mathcal{M}$, we want to find $\nu \subseteq \mathcal{J}$ that maximizes $\sum_{j \in \nu} \pi_j$ subject to $\alpha(i,\nu) \leq \omega$. This problem is the robust binary knapsack problem, which is an $\mathcal{NP}$-hard problem. Hence, suppose instead that we can compute in polynomial time a solution $\nu^*$ with reduced cost $\bar{c}^*$ such that $\alpha(i,\nu^*) \leq 2\omega$ and such that no solution with a smaller reduced cost exists where $\alpha(i,\nu) \leq \omega$. We obtain a relaxed primal solution that may use variables $\lambda_{i\nu}$ with $\alpha(i,\nu)$ up to $2\omega$, and whose objective value is not greater than the optimal value of a linear program where all variables $\lambda_{i\nu}$ have $\alpha(i,\nu) \leq \omega$. As a result, a positive value on the objective function ensures that $FP(\omega)$ is infeasible while a null value provides a fractional feasible solution for $FP(2\omega)$.

It remains to show how to find the solution $\nu^*$. Remark that if $\hat{u} = 0$ (the problem is deterministic), such a solution $\nu^*$ can be found by using the greedy algorithm for the knapsack problem and rounding up the unique fractional variable. Then, one readily verifies that the deterministic approach can be extended to the robust context by applying Theorem 3. □

In the remainder of the section, we let $\omega$ be the solution returned by Theorem 14 and $\lambda^*$ be the corresponding fractional vector. Our objective is to use randomized rounding to obtain an integer solution to $R||U^{\Gamma}|C_{\max}$ with an average makespan of at most $O(\log(m))\omega$. Since $\omega/2$ is a lower bound for opt, this will lead to an average $O(\log(m))$-approximation ratio for $R||U^{\Gamma}|C_{\max}$ (see Theorem 16).

The proposed rounding procedure iteratively adds schedules to all machines until every job is assigned to one of the machines. At each iteration, one additional schedule is selected for each machine and added to the current solution, allowing that the same schedule is added more than once to a given machine. The procedure maintains a variable $y_{ij}$ for each machine $i$ and each job $j$ representing the number of times that job $j$ belongs to a schedule that is added to machine $i$. These variables are used only to prove the approximation bound on the obtained makespan. The integer solution consists of simply assigning each job $j$ to the machine that receives the first schedule that contains $j$.

**Algorithm 2** Randomized rounding (input: a feasible solution ($\lambda^*$) of $FP(\omega)$)

```
y ← 0;
while there exists a job j ∈ J not assigned to any machine do
  for i ← 1, ..., m do
    Randomly select a schedule ν* for machine i with probability λ*_iν of selecting each
    schedule ν;
    for each j ∈ ν* do
      y_ij ← y_ij + 1;
      if job j is not assigned to any machine then
        Assign job j to machine i;
      end if
    end for
  end for
end while
```

The pseudocode for this rounding procedure is given in Algorithm 2. Let $C_{max}$ be the random variable corresponding to the makespan of the schedule computed by Algorithm 2. Let $t$ be the number of iterations performed by the while loop of this algorithm. Since every schedule $\nu$ associated to a variable $\lambda_{i\nu}$ has a total processing time of at most $\omega$, it is clear that $C_{max} \leq \omega t$. Thus, it remains to give an upper bound on the expected value of $t$. For that, we use the well-known Chernoff bound that can be described as follows. Given $K$ independent random variables $X_1, \ldots, X_K$, each one taking the value 1 with certain probability and zero otherwise, such that the expected value of $X = \sum_{k=1}^{K} X_k$ is equal to $\mu$, the probability that $X < (1-\delta)\mu$, for any $\delta > 0$, is smaller than $e^{-\mu\delta^2/2}$. The next Theorem uses this bound to limit the value of $t$.

**Theorem 15.** *The probability that $t > \lceil 4\ln(2n) \rceil$ is less than $1/2$.*

*Proof.* Let $t^* = \lceil 4\ln(2n) \rceil$. For a given job $j$, machine $i$ and iteration $q \leq t^*$ of the while loop, let $X^j_{q,i} = 1$ if the value of $y_{ij}$ is increased during this iteration, and zero otherwise (if the algorithm stopped after $t < t^*$ iterations of the while loop then all the $X^j_{q,i}$ with $t < q \leq t^*$ are set to 0). Clearly, the random variables $X^j_{q,i}$ are independent. Moreover, the constraints (4.12) ensure that $\mathbb{E}(\sum_i X^j_{q,i}) = 1$ for any $j$ and $q$, and thus the expected value of $X^j = \sum_{q=1}^{t^*} \sum_{i \in \mathcal{M}} X^j_{q,i}$ is equal to $t^*$. Now, applying the Chernoff bound with $\delta = 1 - 1/t^*$, and assuming that $t^* \geq 4$, we obtain that

$$\Pr[X^j < 1] < e^{-\frac{(t^*-1)^2}{2t^*}} < e^{-t^*/4} \leq \frac{1}{2n}. \tag{4.16}$$

Note that $\Pr[X^j < 1]$ is the probability that the job $j$ is not scheduled after $t^*$ iterations, and that the random variables $X^1, \ldots, X^n$ are not necessarily independent. Let $X = 1$ if every job is scheduled after $t^*$ iterations, and zero if at least one job is not scheduled. Note that $X = 0$ is equivalent to state that the Algorithm 2 does not finish after $t^*$ iterations, i.e., $t > t^*$. Moreover, we have that

$$\Pr[X = 0] = \Pr[\sum_{j=1}^{n} X^j < n] \leq \sum_{j=1}^{n} \Pr[X^j < 1] < 1/2, \tag{4.17}$$

which completes our proof. □

**Corollary 5.** $\mathbb{E}(C_{max}) = O(\log(n))\omega$.

*Proof.* An immediate consequence of Theorem 15 is that for any integer $c \geq 1$, the probability that $t > c \times \lceil 4\ln(2n) \rceil$ is less than $1/2^c$ (indeed, $1/2^c$ upper bounds the probabilty that none of $c$ parallel execution of Algorithm 2 schedules all the jobs, where each run only performs $\lceil 4\ln(2n) \rceil$ iterations of the while loop). As a result, the expected value of $t$ is smaller than $2 \times \lceil 4\ln(2n) \rceil$. □

A tighter analysis of the approximation ratio of Algorithm 2 yields the following result. Its proof is ommitted for brevity.

**Lemma 9.** $\mathbb{E}(C_{max}) = O(log(m))\omega$.

We obtain easily the following result.

**Theorem 16.** *There is a $O(log(m))$-approximation in expectation for $R||\mathcal{U}^\Gamma|C_{\max}$.*

*Proof of Theorem 16.* Let us define a randomized $O(\log m)$-dual approximation that given a threshold $\omega$ either creates a schedule with $\mathbb{E}(C_{\max}) \leq O(\log m)\omega$, or fails, implying that $\omega < \text{opt}$ (where opt is the optimal solution cost of the $R||U^\Gamma|C_{\max}$ input). Given $\omega$, we apply Theorem 14 to either compute a fractional solution of cost $2\omega$ of $LB$, or fail (implying $\omega < \text{opt}(LB) \leq \text{opt}$). If the algorithm does not fail, we apply Lemma 9 to round this solution to an integer solution with expected makespan $\mathbb{E}(C_{\max}) \leq O(\log m)2\omega$. □

# Chapter 5

# Lot-sizing

*The chapter is based on Santos et al. [2016a].*

## 5.1 Introduction

We consider in this chapter a simple version of the robust lot-sizing problem, denoted $\mathcal{U}^\Gamma$-$LSP$. Consider a finite planning horizon $\{1,\dots,n\}$. For each time period $i = 1,\dots,n$, we are given a capacity $C_i$, a holding cost $p_i$, a shortage cost $s_i$, and a production cost $c_i$. We assume that the demands are not known with precision and vary around their nominal values. However, unlike the previous chapters, it is important here to represent upward deviations as well as downward deviations, because a demand lower than expected may also lead to additional costs (holding costs) Hence, we assume here that the vector of demands $u$ belongs to

$$\mathcal{U}^\Gamma_\pm := \left\{ u \in \mathbb{R}^n : u_i = \overline{u}_i + \xi_i \hat{u}_i, \xi_i \in \{-1,0,1\}, \sum |\xi_i| \leq \Gamma \right\},$$

$\mathcal{U}^\Gamma_\pm$, where $\overline{u}_i$ represents the nominal demand in period $i$, and $\hat{u}_i$ 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.

The problem can be modeled as follows. Variable $x_i$ represents the amount produced at period $i$ and variable $\theta$ represents the total storage and backlog costs. The formulation for $\mathcal{U}^\Gamma$-$LSP$ follows.

$$\min \Big\{ \quad c^T x + \theta \ :$$

$$0 \leq x_i \leq C_i, \quad i = 1,\dots,n, \tag{5.1}$$

$$\theta \geq \sum_{i=1}^{n} \max \left\{ s_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right), -p_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right) \right\}, \quad u \in \mathcal{U}^\Gamma_\pm \Bigg\}. \tag{5.2}$$

Let us make a few comments on the above formulation. First, we remark that $\mathcal{U}^\Gamma$-$LSP$ is not an instance of $\mathcal{U}^\Gamma$-$CO$ because variables $x$ are fractional. This being said, the algorithms developed in the balance of this chapter, like those developed in Chapters 3 and 4 rely on the combinatorial structure of $\mathcal{U}^\Gamma$. Then, the attentive reader will also notice that constraints (5.2) are non-linear, unlike the linear constraints (1.4). We do not linearize these constraints purposely as their non-linearity shall be exploited in this chapter when devising separation algorithms. Finally, we point out that the formulation is not compatible with variable uncertainty because $x$ is fractional. Nevertheless, even if $x$ were binary, the nature of the algorithms proposed in this chapter would not suit variable uncertainty, because these algorithms generate a subset of constraints (5.2) that must be valid for all $x$. We come back to this in the next chapter.

We show in the next section how to solve $\mathcal{U}^\Gamma$-$LSP$ through a row-and-column generation algorithm. Section 5.3 focuses on the separation problem and shows how it can be solved in pseudo-polynomial time through dynamic programming. For simplicity, the section focuses on the simpler uncertainty set $\mathcal{U}^\Gamma$. Section 5.4 is devoted to construct a Fully Polynomial Time Approximation Scheme for the problem (FPTAS). Computational experiments are presented in Section 5.5.

## 5.2 Decomposition algorithm

The problem $\mathcal{U}^\Gamma$-$LSP$ contains exponential numbers of variables and constraints, making it intractable as such. Here, we tackle the problem by generating a relevant subset of variables and constraints on the fly in the course of the decomposition algorithm presented below. Let $\mathcal{U}^* \subset \mathcal{U}^\Gamma_\pm$ be a finite set. We can reformulate

$$\theta \geq \sum_{i=1}^{n} \max\left\{ s_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right), -p_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right) \right\}, \quad u \in \mathcal{U}^*, \tag{5.3}$$

as the following set of linear inequalities, written for each $u \in \mathcal{U}^*$:

$$\theta \geq \sum_{i=1}^{n} \varphi_i^u, \tag{5.4}$$

$$\varphi_i^u \geq s_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right), \quad i = 1, \ldots, n, \tag{5.5}$$

$$\varphi_i^u \geq -p_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right), \quad i = 1, \ldots, n, \tag{5.6}$$

where $\varphi^u$ is an additional vector of optimization variables. Our approach is based on the above linear reformulation of (5.3). Specifically, we relax constraints (5.2) for all elements

in $\mathcal{U}_{\pm}^{\Gamma}$ but $\mathcal{U}^*$ and replace robust constraint (5.2) by its linear reformulation, obtaining the relaxed Master Problem

$$
\begin{array}{ll}
\min \Big\{ & c^T x + \theta \ : \\
& 0 \leq x_i \leq C_i, \quad i = 1, \ldots, n, \\
\mathcal{U}^*\text{-}LSP \qquad & \theta \geq \sum_{i=1}^{n} \varphi_i^u, \quad u \in \mathcal{U}^*, \\
& \varphi_i^u \geq s_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right), \quad u \in \mathcal{U}^*, i = 1, \ldots, n, \\
& \varphi_i^u \geq -p_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j \right), \quad u \in \mathcal{U}^*, i = 1, \ldots, n \quad \Big\}.
\end{array}
$$

Given a feasible solution $(x^*, \theta^*)$ to $\mathcal{U}^*$-$LSP$, one checks the feasibility of $(x^*, \theta^*)^*$ for $\mathcal{U}^{\Gamma}$-$LSP$ by solving the adversarial problem

$$
\max_{u \in \mathcal{U}_{\pm}^{\Gamma}} \sum_{i=1}^{n} \max \left\{ s_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j^* \right), -p_i \left( \sum_{j=1}^{i} u_j - \sum_{j=1}^{i} x_j^* \right) \right\}; \tag{5.7}
$$

we denote the objective function of (5.7) as $g(u)$ for short. Let $u^*$ be the optimal solution for the adversarial problem. If $g(u^*) > \theta^*$, then $\mathcal{U}^* \leftarrow \mathcal{U}^* \cup \{u^*\}$, and the corresponding optimization vector $\varphi^{u^*}$ and constraints (5.4)–(5.6) are added to $\mathcal{U}^*$-$LSP$. Hence, unlike the cutting plane algorithm depicted in Section 1.1, the algorithm presented above is a row-and-column generation algorithm.

## 5.3 Dynamic programming algorithm

For the sake of simplicity, we first restrict ourselves to upward deviations only (resprresented by $\mathcal{U}^{\Gamma} \subseteq \mathcal{U}_{\pm}^{\Gamma}$), showing at the end of the section how the assumption can be relaxed. Let $u_i^{\Sigma} = \sum_{j=1}^{i} u_j$ define the sum of the $i$ first elements of $u$. Then, we define $f_i(u_i^{\Sigma}) = \max \left\{ s_i \left( u_i^{\Sigma} - \sum_{j=1}^{i} x_j^* \right), -p_i \left( u_i^{\Sigma} - \sum_{j=1}^{i} x_j^* \right) \right\}$, for each $i = 1, \ldots, n$ and $f(u^{\Sigma}) = \sum_{i=1}^{n} f_i(u_i^{\Sigma})$. We also denote the function by $f(u^{\Sigma}, x)$ when the dependency on $x$ is needed.

We are interested here in solving the optimization problem

$$
AP \qquad \max_{u \in \mathcal{U}^{\Gamma}} f(u^{\Sigma}). \tag{5.8}
$$

Problem (5.8) may not be easy to solve since $\mathcal{U}^{\Gamma}$ contains an exponential number of elements and function $f$ is non-linear. However, recall that, because of the definition of $\mathcal{U}^{\Gamma}$, we do not need to know the entire vector $u \in \mathcal{U}^{\Gamma}$ to compute $f(u^{\Sigma})$. In fact,

it is enough to know the cumulative uncertainties $u_i^\Sigma = \sum_{j=1}^{i} u_j$ for each $i = 1, \ldots, n$, which are equivalent to the cumulative deviations $\sum_{j=1}^{i} u_j - \sum_{j=1}^{i} \overline{u}_j$ for each $i = 1, \ldots, n$ because $u \in [\overline{u}, \overline{u} + \hat{u}]$. With this in mind, we introduce

$$f_i'(\phi_i) = \max \left\{ s_i \left( \bar{u}_i^\Sigma + \phi_i - \sum_{j=1}^{i} x_j^* \right), -p_i \left( \bar{u}_i^\Sigma + \phi_i - \sum_{j=1}^{i} x_j^* \right) \right\},$$

obtained from $f_i$ by treating separately the cumulative mean $\bar{u}_i^\Sigma = \sum_{j=1}^{i} \overline{u}_j$ and the cumulative deviation $\phi_i = u_i^\Sigma - \bar{u}_i^\Sigma$. Namely, let $\xi \in \{0,1\}^n$ be a binary vector that satisfies $\|\xi\|_1 \leq \Gamma$ and let $u \in \mathcal{U}^\Gamma$ be the associated vector of uncertain parameters, defined as $u_i = \overline{u}_i + \xi_i \hat{u}_i$ for each $i = 1, \ldots, n$. One readily checks that $f_i(u_i^\Sigma) = f_i'(\phi_i)$ if and only if $\phi_i = \sum_{j=1}^{i} \hat{u}_j \xi_j$. Therefore, adversarial problem (5.8) can be rewritten as

$$AP \qquad \max \Bigg\{ \sum_{i=1}^{n} f_i'(\phi_i) \ : \ \phi_i = \sum_{t=1}^{i} \hat{u}_t \xi_t, \quad i = 1, \ldots, n, \ \sum_{i=1}^{n} \xi_i \leq \Gamma, \ \xi_i \in \{0,1\}, \quad i = 1, \ldots, n \Bigg\}.$$

Up to now we have shown that the optimal solution cost of $AP$ only depends on the cumulative deviations $\phi_i$ for each $i = 1, \ldots, n$. To obtain a Dynamic Programming Algorithm (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,

$$\overline{\phi} = \max_{S \subseteq \{1,\ldots,n\} : |S| = \Gamma} \sum_{i \in S} \hat{u}_i.$$

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 $\gamma$ deviations

and a cumulative deviation of $\phi$:

$$\begin{aligned}
\alpha(j,\gamma,\phi) \quad = \quad \max\Big\{ \quad & \sum_{i=1}^{j} f_i'(\phi_i) \ : \\
& \phi_j = \phi, && (5.9)\\
& \phi_i = \sum_{t=1}^{i} \hat{u}_t \xi_t, \quad i = 1,\ldots,j, && (5.10)\\
& \sum_{i=1}^{j} \xi_i \le \gamma, && (5.11)\\
& \xi_i \in \{0,1\}, \quad i = 1,\ldots,j \quad \Big\}. && (5.12)
\end{aligned}$$

Let $\alpha(j,\gamma,\phi) = -\infty$ if the feasible set defined by $(5.9)-(5.12)$ is empty because the value of $\phi$ cannot be reached by a sum of deviations. Hence, we have that $\alpha(1,0,0) = f_1'(0)$, $\alpha(1,\gamma,\hat{u}_1) = f_1'(\hat{u}_1)$ for each $1 \le \gamma \le \Gamma$, 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 $\text{opt}(AP)$, can be computed as $\text{opt}(AP) = \max_{\phi=0,\ldots,\overline{\phi}} \alpha(n,\Gamma,\phi)$. Moreover, we see easily by contradiction that $\alpha(n,\gamma,\phi)$ satisfies the functional equation stated below.

**Lemma 10.** *For $j > 1$, each $\alpha(j,\gamma,\phi)$ can be obtained using the following recursion:*

$$\begin{aligned}
&\alpha(j,\gamma,\phi) = f_j'(\phi) + \max\{\alpha(j-1,\gamma,\phi), \alpha(j-1,\gamma-1,\phi-\hat{u}_j)\},\\
&\alpha(j,0,\phi) = f_j'(\phi) + \alpha(j-1,0,\phi)
\end{aligned} \qquad (5.13)$$

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

It follows from the equivalence between separation and optimization [Grötschel et al., 1993] and the cutting plane algorithm presented in the corollary below that $\mathcal{U}^\Gamma$-$LSP$ can be solved in pseudo-polynomial time.

**Corollary 6.** *Problem $\mathcal{U}^\Gamma$-$LSP$ can be solved in pseudo-polynomial time.*

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

$$\begin{aligned}
& \min\Big\{ \quad c^T x + \theta \ : \\
\mathcal{U}^*\text{-}LSP' \qquad & 0 \le x_i \le C_i, \quad i = 1,\ldots,n,\\
& \sum_{i=1}^{n} \mathbf{f}_i^j(x) \le \theta, \quad j = 1,\ldots,J \quad \Big\},
\end{aligned}$$

initialized with $J = 0$. Given a feasible solution $x^*$ to $\mathcal{U}^*$-$LSP'$, we solve the adversarial problem. If $\max_{u \in \mathcal{U}^\Gamma} f(u^\Sigma, x^*) > \theta^*$, we let $u^*$ be an optimal solution of the maximization problem, set $J \leftarrow J+1$, and add a new constraint to $\mathcal{U}^*$-$LSP'$ where $\mathbf{f}_i^j(x) = s_i \left(u_i^\Sigma - \sum_{j=1}^i x_j\right)$ if $s_i \left(u_i^\Sigma - \sum_{j=1}^i x_j\right) = \max\left\{s_i \left(u_i^\Sigma - \sum_{j=1}^i x_j^*\right), -p_i \left(u_i^\Sigma - \sum_{j=1}^i x_j^*\right)\right\}$ and $\mathbf{f}_i^j(z) = -p_i \left(u_i^\Sigma - \sum_{j=1}^i x_j^*\right)$, otherwise. □

Lemma 11 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 $\Gamma$ is constant since $\phi$ can take at most $n^\Gamma$ different values.

We conclude the section by showing that downward deviations of $u$ can be handled by replacing constraints (5.11) and (5.12) 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 (5.13) is then adapted to:

$$\alpha(j, \gamma, \phi) = f'_j(\phi) + \max\{\alpha(j-1, \gamma, \phi), \alpha(j-1, \gamma-1, \phi - \hat{u}_j), \alpha(j-1, \gamma-1, \phi + \hat{u}_j)\},$$
$$\alpha(j, 0, \phi) = f'_j(\phi) + \alpha(j-1, \gamma, \phi),$$

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

## 5.4 Fully polynomial time approximation scheme

We show next how to modify our DPA to obtain a FPTAS for problems $\mathcal{U}^\Gamma$-$LSP$ that satisfy additional assumptions. For simplicity, the FPTAS is exposed for upward deviations only. One readily extends it to account for downward deviations as in the end of the previous section. Our approach works in two steps. First, we adapt to $AP$ the FPTAS proposed for the knapsack problem by Ibarra and Kim [1975]. 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 parameters $s_i$ and $p_i$ satisfy the technical assumption stated below. Then, we show that a FPTAS for $AP$ can be turned into a FPTAS for $\mathcal{U}^\Gamma$-$LSP$. Consider the following lower bound for the optimal solution of the problem

$$LB = \frac{\boldsymbol{u} \min\{s_n, p_n\}}{2},$$

where $\boldsymbol{u} = \max_{i=1,\ldots,n} \hat{u}_i$.

**Assumption 1.** *We suppose that* $\min\{s_n, p_n\} > 0$ *and that*

$$\frac{2 \max\limits_{i=1,\dots,n} \{s_i, p_i\}}{\min\{s_n, p_n\}} \leq \mathcal{P}(n) \tag{5.14}$$

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

When $\mathcal{P}(n)$ is equal to some constant $\lambda > 0$, we obtain the set of instances for which $\frac{\max\limits_{i=1,\dots,n} \{s_i, p_i\}}{\min\{s_n, p_n\}} \leq \frac{\lambda}{2}$. Considering polynomials of higher degrees yields a larger set of admissible instances while increasing the computational complexity of the resulting FPTAS.

**Lemma 12.** *Consider problem AP such that Assumption 1 holds. There exists a FPTAS for AP.*

*Proof.* For any $\epsilon > 0$, we let $K = \frac{\epsilon \boldsymbol{u} \min\{s_n, p_n\}}{4n(\Gamma+1)\left|\max\limits_{i=1,\dots,n} \{s_i, p_i\}\right|}$ and define the new mean value $\overline{u}'_i = \frac{\overline{u}_i}{K}$ and deviation $\hat{u}'_i = \lfloor \frac{\hat{u}_i}{K} \rfloor$ for each $i = 1, \dots, n$, and $\overline{\phi}' = \max_{S \subseteq \{1,\dots,n\}:|S|=\Gamma} \sum_{i \in S} \hat{u}'_i$. Then, execute the DPA presented in the previous section to $AP$ using the vector of deviations $\hat{u}'$. Using notation $\boldsymbol{u}' = \lfloor \frac{\boldsymbol{u}}{K} \rfloor$, we see that the running time of the algorithm is polynomial in $(n, \Gamma, 1/\epsilon)$ since

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

We are left to show that the optimal solution to the problem with $\hat{u}'$ 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 $\text{profit}(\cdot)$ and $\text{profit}'(\cdot)$ denote the profit of any element of $\{0,1\}^n$ using deviations $\hat{u}$ and $\hat{u}'$, respectively. Clearly, $\text{profit}(\xi') \leq \text{opt}(AP)$. Then, recall from the definition that $K\,\text{profit}'(\xi) =$

$$\sum_{i=1}^{n} \max\left\{ -s_i K \sum_{t=1}^{i} x_i^* + s_i \sum_{t=1}^{i} \left(\overline{u}_t + \xi_t K \left\lfloor \frac{\hat{u}_t}{K} \right\rfloor\right), p_i K \sum_{t=1}^{i} x_i^* - p_i \sum_{t=1}^{i} \left(\overline{u}_t + \xi_t K \left\lfloor \frac{\hat{u}_t}{K} \right\rfloor\right) \right\},$$

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

$$|\,\text{profit}(\xi) - K\,\text{profit}'(\xi)| \leq n(\Gamma+1)K \left| \max_{i=1,\dots,n} \{s_i, p_i\} \right|. \tag{5.15}$$

Therefore,

$$\begin{aligned}\text{profit}(\xi') &\geq K\,\text{profit}'(\xi') - n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{s_i,p_i\}\right| \\ &\geq K\,\text{profit}'(\xi^*) - n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{s_i,p_i\}\right| \\ &\geq \text{profit}(\xi^*) - 2n(\Gamma+1)K\left|\max_{i=1,\ldots,n}\{s_i,p_i\}\right| \\ &= \text{opt}(AP) - \epsilon LB \geq (1-\epsilon)\,\text{opt}(AP),\end{aligned}$$

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 $\mathcal{U}^\Gamma$-$LSP$.

**Lemma 13.** *Assume that there exists a FPTAS for* $\max_{u\in\mathcal{U}^\Gamma} f(u^\Sigma)$. *There exists a FPTAS for* $\mathcal{U}^\Gamma$-$LSP$.

*Proof.* We must show that for each $\epsilon > 0$, we can provide in polynomial time an $(1+\epsilon)$-approximate solution to $\mathcal{P}$. Our approach relies on the cutting-plane algorithm from Corollary 6 with the difference that each $AP$ is now solved with the FPTAS to provide an $\frac{1}{1+\epsilon}$-approximate solution. Let $(x', \theta')$ be the solution returned by the approximate cutting plane algorithm. We claim that $(x', (1+\epsilon)\theta')$ is the desired approximate solution. Clearly, $(x', (1+\epsilon)\theta')$ is computed in polynomial time. Then, we must verify that

$$\text{opt}(\mathcal{U}^\Gamma\text{-}LSP) \leq c^T x' + (1+\epsilon)\theta' \leq (1+\epsilon)\,\text{opt}(\mathcal{U}^\Gamma\text{-}LSP).$$

To prove the first inequality, we rewrite $\mathcal{U}^\Gamma$-$LSP$ as

$$\min_{0\leq x\leq C} c^T x + F(x),$$

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

$$c^T x' + (1+\epsilon)\theta' \geq c^T x' + F(x') \geq \text{opt}(\mathcal{U}^\Gamma\text{-}LSP).$$

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

$$\text{opt}(\mathcal{U}^\Gamma\text{-}LSP) < \frac{c^T x'}{1+\epsilon} + \frac{1+\epsilon}{1+\epsilon}\theta' \leq c^T x' + \theta'. \tag{5.16}$$

Moreover, because $\theta'$ is an approximate solution of the corresponding $AP$, we have

$$c^T x' + \theta' \leq c^T x' + F(z') \leq \text{opt}(\mathcal{U}^\Gamma\text{-}LSP),$$

which is in contradiction with (5.16). □

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

Table 5.1: Values of $\Gamma^\epsilon$ obtained by rounding up the values prescribed by the probabilistic bound $\beta(\Gamma, n)$ from (1.11).

## 5.5 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.

### 5.5.1 Instances and details

We consider three numbers of periods: 50, 100, and 200. For each 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 $\Gamma$, motivated by the probabilistic bounds given in Bertsimas and Sim [2004] and provided in Table 5.1. Namely, the value of $\Gamma^\epsilon$ is such that there is a probability of $1 - \epsilon$ that the real cost will be no higher than the optimal solution cost whenever $\xi$ is composed of independent and identically distributed random variables.

Our experiments compare the DPA with the well-known MIP reformulation of $AP$

| $n$ | 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 5.2: Arithmetic means of the solution times.

(e.g. Bienstock and Özbay [2008]) recalled below:

$$
\begin{aligned}
\max \Big\{ \quad & \sum_{i=1}^{n} \varphi_i \ : \\
& \varphi_i \le s_i \left( \sum_{j=1}^{i} x_i - y_i \right) + M_i w_i, \qquad i = 1, \dots, n, \\
& \varphi_i \le -p_i \left( \sum_{j=1}^{i} x_i - y_i \right) + M_i (1 - w_i), \qquad i = 1, \dots, n, \\
& y_i = \sum_{i=1}^{i} (\overline{u}_i + \hat{u}_i \xi_i), \qquad i = 1, \dots, n, \\
& \sum_{i=1}^{n} \xi_i \le \Gamma, \qquad i = 1, \dots, n, \\
& \xi \in \{0,1\}^n, \ w \in \{0,1\}^n, \ y \ge 0 \quad \Big\}.
\end{aligned}
$$

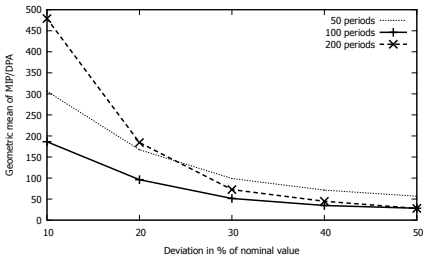
For each $i = 1, \dots, n$, variables $\varphi_i$ and $y_i$ represent the value of $f_i(y_i(\xi))$ and $u_i^{\Sigma}$, respectively.

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 CPLEX [2013]. The numerical experiments were carried out in an Intel(R) Core(TM) i7 CPU M60, 2.6Hz 4GB Ram machine.

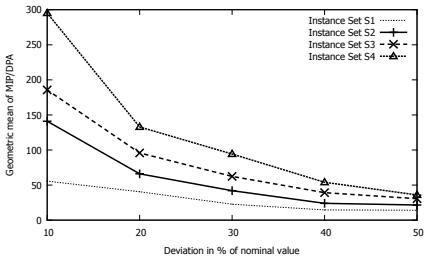
### 5.5.2 Results

We provide in Figure 5.1 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 5.2 for both approaches. We present on Table 5.2 the arithmetic means of the solution times for the different number of time periods and levels of deviations.

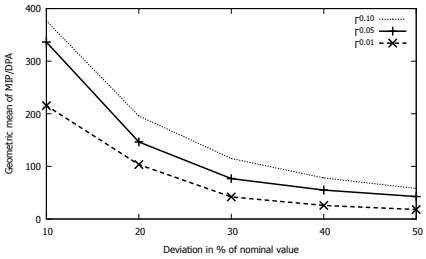
Figure 5.1 shows that DPA clearly outperforms MIP, with means ranging up to 475 when $n = 200$ and the deviation is 10%. The different parameters strongly impact the respective solution times. First, we see on the charts from Figure 5.1 that, as expected from its theoretical complexity, higher levels of deviation slow down DPA. The number of




(a) Varying the number of time periods.

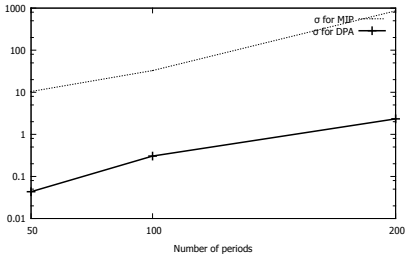



(b) Different sets of instances.

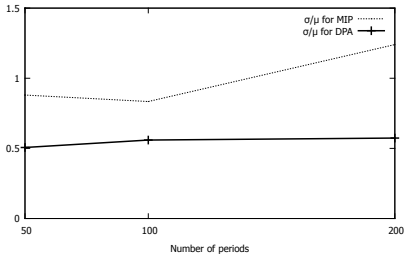



(c) Different values of Γ.

Figure 5.1: Geometric mean of (time MIP)/(time DPA).




(a) Standard deviation represented in logarithmic scale.




(b) Standard deviation divided by the arithmetic mean ($\mu$).

Figure 5.2: Standard deviation of the solution times $\sigma$ when varying the number of time periods.

time periods strongly affect both approaches, see Table 5.2. When the deviation is small, Figure 5.1(a) 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 5.1(b) 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 5.1(b). Finally, Figure 5.1(c) 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 5.2 for DPA and MIP, through the standard deviation ($\sigma$). Figure 5.2(a) 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 5.2(b) shows these standard deviations in a relative scale, dividing them by the associated arithmetic means. The figure shows that in a relative scale, MIP varies roughly twice as much as DPA.

To conclude, 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, we get back to this in Chapter 7.

# Chapter 6

# Vehicle routing with time windows

*The chapter is based on Agra et al. [2013], Santos et al. [2016a].*

## 6.1 Introduction

We consider in this chapter a vehicle routing problem with time windows, which can be defined as follows. We are given a directed graph $G = (N, A)$, a set of vehicles $K$, a cost function $\kappa : A \times K \to \mathbb{R}_+$, and a vector of traveling times $(u^1, \ldots, u^K)$ where each subvector $u^k$ may take any value in $\mathcal{U}^\Gamma \subset \mathbb{R}_+^{|A|}$ (hence $u \in \prod_{k \in K} \mathcal{U}^\Gamma$). The graph contains special depot nodes $o$ (origin) and $d$ (destination) connected to all other nodes of $G$, and we denote by $N^*$ the set of nodes that are not depots, $N^* := N \backslash \{o, d\}$. We are given time windows $[\underline{b}_i, \overline{b}_i]$ with $\underline{b}_i, \overline{b}_i \in \mathbb{R}_+$, for each $i \in N^*$. Because different vehicles may have access to different routes, we also introduce the subset $A^k$ of $A$ for each $k \in K$. Problem $\mathcal{U}^\Gamma$-$VRPTW$ consists of defining routes for the vehicles in $K$ such that the union of all routes passes exactly once by each $i \in N^*$. Specifically, the problem can be formulated with a set of binary flow variables $x_{ij}^k$ which indicate whether vehicle $k$ travels from node $i \in N$ to node $j \in N$. Variables $x$ must satisfy

$$\sum_{k \in K} \sum_{j \in N:(i,j) \in A^k} x_{ij}^k = 1, \qquad i \in N^*, \tag{6.1}$$

$$\sum_{j \in N:(j,i) \in A^k} x_{ji}^k - \sum_{j \in N:(i,j) \in A^k} x_{ij}^k = \begin{cases} -1 & i = o \\ 1 & i = d \\ 0 & otherwise \end{cases}, \qquad i \in N,\ k \in K, \tag{6.2}$$

where constraints (6.1) ensure that all $i \in N^*$ are served exactly once, and constraints (6.2) are the flow conservation constraints for each vehicle. At this point, $x$ could form cycles not connected to $o$ nor $d$, which must be prevented by adding cycle-breaking inequalities to the formulation. We do not detail them further here as the time windows constraints, detailed below, will prevent cycles from appearing.

Different variants of the problem can be defined, depending on the type of time windows that must be satisfied by the routes. Namely, left and right time windows can be considered as soft or hard: a soft time window means that a violation is allowed at some cost, assumed proportional to the extend of the violation, while a hard time window must be satisfied. In particular, a vehicle arriving at a node before a hard left time window must wait before actually entering the node, while a soft left time windows allows the vehicle to enter the node before the the window starts. We disregard in this chapter the configuration where the vehicle must wait if it arrives before the left time window and pays for the waiting time, which can be handled using algorithms similar to those described in the following. To handle time windows, the above formulation must be complemented by introducing a set of continuous variables $y_i^{ku}$ indicating the arrival time of vehicle $k$ at node $i \in N$ for traveling times $u \in \prod_{k\in K} \mathcal{U}^\Gamma$. In fact, since only one vehicle can serve node $i$, we may drop the index $k$ and let $y_i$ be the arrival time at node $i$ of the vehicle that serves $i$. Variables $x$ and $y$ are linked through quadratic constraints

$$x_{ij}^k(y_i^u + u_{ij}^k - y_j^u) \leq 0, \qquad (i,j) \in A^k, k \in K, u \in \prod_{k\in K} \mathcal{U}^\Gamma, \tag{6.3}$$

and intialized through

$$y_o^u = 0, \quad u \in \prod_{k\in K} \mathcal{U}^\Gamma. \tag{6.4}$$

Then, for each $i \in N^*$ and $u \in \prod_{k\in K} \mathcal{U}^\Gamma$ hard time windows are modeled by

$$\underline{b}_i \leq y_i^u \qquad \text{and} \qquad y_i^u \leq \overline{b}_i, \tag{6.5}$$

while soft time windows are modeled by introducing additional variable $\varphi_i^u$ representing the cost for the earliness and tardiness for vehicle $k$

$$\varphi_i^{ku} \geq s_i^k \max(\underline{b}_i - y_i^u, 0) \qquad \text{and} \qquad \varphi_i^{ku} \geq p_i^k \max(y_i^u - \overline{b}_i, 0), \tag{6.6}$$

where $s$ and $p$ are vectors describing the unitary earliness and tardiness costs, respectively. Notice that, unlike variables $y$, variables $\varphi$ must depend on the vehicle $k$ because the earliness and tardiness costs are dependant on vehicle $k$ for each $k \in K$. Finally, the objective of $\mathcal{U}^\Gamma$-$VRPTW$ is given by

$$\sum_{k\in K} \sum_{(i,j)\in A^k} \kappa_{ij}^k x_{ij}^k,$$

possibly adding the sum of earliness and tardiness costs

$$\max_{u\in\prod_{k\in K}\mathcal{U}^\Gamma} \sum_{i\in N} \sum_{k\in K} \varphi_i^{ku}. \tag{6.7}$$

We show next how to linearize efficiently constraints (6.3). Using big-$M$ coefficients, we obtain the classical reformulation

$$y_i^u - y_j^u + (u_{ij}^k + M)x_{ij}^k \leq M, \qquad (i,j) \in A^k,\ k \in K, u \in \prod_{k\in K} \mathcal{U}^\Gamma. \tag{6.8}$$

Since in any feasible solution, at most one vehicle will take arc $(i,j)$, (6.8) can be strengthened to

$$y_i^u - y_j^u + \sum_{k\in K:(i,j)\in A^k} (u_{ij}^k + M)x_{ij}^k \leq M, \qquad (i,j)\in A, u \in \prod_{k\in K} \mathcal{U}^\Gamma. \tag{6.9}$$

The simple strengthening procedure used from (6.8) to (6.9) was originally introduced in Agra et al. [2013] and it has proved its efficiency for other (non-robust) routing problems, see for instance Agra et al. [2014].

Finally, we explain below how the uncertainty set $\prod_{k\in K}\mathcal{U}^\Gamma$ can be replaced by an uncertainty set of smaller cardinality. Given two different vehicles $k \neq h$, the travel times $u^k$ and $u^h$ are independent, because the considered uncertainty set is defined by the Cartesian product of two copies of $\mathcal{U}^\Gamma$, one for $k$ and one for $h$. However, it is not necessary to consider all combinations of $u^k$ and $u^h$ in $\mathcal{U}^\Gamma \times \mathcal{U}^\Gamma$. Specifically, we see that constraints (6.9) are equivalent to

$$y_i^u - y_j^u + \sum_{k\in K:(i,j)\in A^k} (M + u_{ij}^k)x_{ij}^k \leq M, \quad (i,j)\in A, u \in \mathcal{U}^\Gamma_{diag}, \tag{6.10}$$

defined for the smaller uncertainty set

$$\mathcal{U}^\Gamma_{diag} := \left\{(u,\ldots,u) \in \mathbb{R}^{|A|\times|K|} : u \in \mathcal{U}^\Gamma\right\}.$$

Similarly, constraints (6.4), (6.5), and (6.6) written for all $u \in \prod_{k\in K}\mathcal{U}^\Gamma$ become

$$y_o^u = 0, \qquad u \in \mathcal{U}^\Gamma_{diag} \tag{6.11}$$
$$\underline{b}_i \leq y_i^u, \qquad i \in N, u \in \mathcal{U}^\Gamma_{diag}, \tag{6.12}$$
$$y_i^u \leq \bar{b}_i, \qquad i \in N, u \in \mathcal{U}^\Gamma_{diag}, \tag{6.13}$$
$$\varphi_i^{ku} \geq s_i^k \max(\underline{b}_i - y_i^u, 0), \qquad i \in N, k \in K, u \in \mathcal{U}^\Gamma_{diag}, \tag{6.14}$$
$$\varphi_i^{ku} \geq p_i^k \max(y_i^u - \bar{b}_i, 0), \qquad i \in N, k \in K, u \in \mathcal{U}^\Gamma_{diag}, \tag{6.15}$$

and the earliness and tardiness costs become

$$\max_{u\in\mathcal{U}^\Gamma_{diag}} \sum_{i\in N}\sum_{k\in K} \varphi_i^{ku}.$$

One direction of the equivalence is immediate since $\mathcal{U}^\Gamma_{diag} \subset \prod_{k\in K}\mathcal{U}^\Gamma$; the other direction follows easily from the fact only one vehicle is involved in each constraint of (6.10).

We detail in the following four sections how the four combinations of hard and soft time windows can be addressed through decomposition algorithms. In each case, dynamic programming algorithms are proposed to handle the separation problems.

## 6.2 Left and right hard

If both time windows are hard, we can strengthen (6.10) by choosing $M = \overline{b}_i - \underline{b}_j$ for each $(i,j) \in A$, obtaining

$$\begin{array}{ll} y_i^u - y_j^u + \sum\limits_{k \in K:(i,j) \in A^k} (\overline{b}_i - \underline{b}_j + u_{ij}^k) x_{ij}^k \leq \overline{b}_i - \underline{b}_j, & (i,j) \in A, u \in \mathcal{U}_{diag}^{\Gamma}, \\ \underline{b}_i \leq y_i^u \leq \overline{b}_i, & i \in N, u \in \mathcal{U}_{diag}^{\Gamma}. \end{array} \tag{6.16}$$

The resulting mixed-integer linear formulation follows, extending to the robust context what is sometime known as the Resource Inequalities formulation

$$\mathcal{U}^{\Gamma}\text{-}RI \qquad \min \left\{ \sum_{k \in K} \sum_{(i,j) \in A^k} \kappa_{ij}^k x_{ij}^k : (6.1), (6.2), (6.11), (6.16), y \geq 0, x \text{ binary} \right\}$$

### 6.2.1 Iterative algorithms

Formulation $\mathcal{U}^{\Gamma}$-$RI$ can be used as such only for very small instances, since the cardinality of $\mathcal{U}_{diag}^{\Gamma}$ increases exponentially with $|A|$. To deal with larger instances, one must devise decomposition algorithms, in the line of the row-and-column generation algorithm described in Chapter 5. Specifically, consider subset $\mathcal{U}^* \subset \mathcal{U}_{diag}^{\Gamma}$ defining the relaxed master problem $\mathcal{U}^*$-$RI$

$$\begin{array}{l} \min \Big\{ \sum\limits_{k \in K} \sum\limits_{(i,j) \in A^k} \kappa_{ij}^k x_{ij}^k : (6.1), (6.2), y \geq 0, x \text{ binary}, y_o^u = 0, \quad u \in \mathcal{U}^*, \\ \quad \underline{b}_i \leq y_i^u \leq \overline{b}_i, \quad i \in N, u \in \mathcal{U}^*, \\ \quad y_i^u - y_j^u + \sum\limits_{k \in K:(i,j) \in A^k} \max\{\overline{b}_i + u_{ij}^k - \underline{b}_j, 0\} x_{ij}^k \leq \overline{b}_i - \underline{b}_j, \ (i,j) \in A, u \in \mathcal{U}^* \Big\}. \end{array}$$

Then, given a feasible solution $x^*$ to $\mathcal{U}^*$-$RI$, we must find out whether there exists $u^* \in \mathcal{U}_{diag}^{\Gamma} \setminus \mathcal{U}^*$ that leads to a violated group of constraints

$$\begin{array}{ll} y_i^{u^*} - y_j^{u^*} + \sum\limits_{k \in K:(i,j) \in A^k} \max\{\overline{b}_i + {u^*}_{ij}^k - \underline{b}_j, 0\} x_{ij}^k \leq \overline{b}_i - \underline{b}_j, & (i,j) \in A, \\ \underline{b}_i \leq y_i^{u^*} \leq \overline{b}_i, & i \in N. \end{array} \tag{6.17}$$

If such a violated group of constraints is found, then $\mathcal{U}^* \leftarrow \mathcal{U}^* \cup \{u^*\}$, and the corresponding optimization vector $y^{u^*}$ and the group of constraints (6.17) are added to $\mathcal{U}^*$-$RI$, yielding a row-and-column generation algorithm. The above algorithm is different from the one depicted in Chapter 5 in two aspects. First, the restricted master problem $\mathcal{U}^*$-$RI$ is now a MILP and is $\mathcal{NP}$-hard to solve exactly even for a single scenario in $\mathcal{U}^*$. Second, we show in Subection 6.2.2 how the separation problem can be solved in strongly polynomial time.

We present next another formulation for the problem, sometimes known as the Path Inequalities formulation. The Path Inequalities formulation does not explicitly consider the satisfaction of the time windows. Instead, it forbids routes in $G$ for which it is not possible to find arrival times that satisfy the time windows. Let $\mathcal{P}^k$ be the set of infeasible paths in $G$ starting at $o$, that is, the set of paths in $G$ for which it is not possible to define arrival times $y$ that satisfy constraints (6.16). For $p \in \mathcal{P}^k$, we denote by $|p|$ the number of arcs contained in $p$. The main idea of formulation $\mathcal{U}^\Gamma$-$\mathcal{P}I$ is to forbid such paths, through paths inequalities

$$\sum_{(i,j)\in p} x_{ij}^k \leq |p| - 1, \quad p \in \mathcal{P}^k,\ k \in K. \tag{6.18}$$

The Path Inequalities formulation $\mathcal{U}^\Gamma$-$\mathcal{P}I$ is

$$\min \left\{ \sum_{k\in K} \sum_{(i,j)\in A^k} \kappa_{ij}^k x_{ij}^k : (6.1), (6.2), (6.18), \text{cycle-breaking inequalities}, x \text{ binary} \right\},$$

where the *cycle-breaking inequalities* could be any set of constraints (possibly with additional variables) that forbid cycles. In our computational results we use the MTZ inequalities [Miller et al., 1960] which, essentially, uses an auxiliary set of variables similar to $y$ in $\mathcal{U}^\Gamma$-$RI$ to impose an order on the nodes visited by the vehicles. As before, $\mathcal{U}^\Gamma$-$\mathcal{P}I$ can hardly be solved as such, unless very small instances are considered. Rather, we should handle the formulation through a cutting plane algorithm that generates constraints (6.18) on the fly. Let $\mathcal{P}^{*k} \subset \mathcal{P}^k$ for each $k \in K$. We define the relaxed master problem $\mathcal{U}^\Gamma$-$\mathcal{P}^*I$ as

$$\min \left\{ \sum_{k\in K} \sum_{(i,j)\in A^k} \kappa_{ij}^k x_{ij}^k : (6.1), (6.2), \text{cycle-breaking inequalities}, x \text{ binary}, \right.$$
$$\left. \sum_{(i,j)\in p} x_{ij}^k \leq |p| - 1, \quad p \in \mathcal{P}^{*k},\ k \in K \right\}.$$

Then, given a feasible solution $x^*$ to $\mathcal{U}^\Gamma$-$\mathcal{P}^*I$, we must find out whether there exists $p^* \in \mathcal{P}^{*k} \setminus \mathcal{P}^k$ and $k \in K$ that leads a violated path inequality constraint, in which case the constraint is added to $\mathcal{U}^\Gamma$-$\mathcal{P}^*I$ and the problem is solved again. In practice, it is more efficient to implement the above algorithm in a branch-and-cut way, inserting the cutting planes at the root and integer nodes in the branch-and-bound tree. An interesting characteristic of formulation $\mathcal{U}^\Gamma$-$\mathcal{P}I$ is that all information on the robust aspects is hidden into (6.18). In particular, the only difference between our solution algorithm and the one proposed by Kallehauge et al. [2007] lies in the separation of violated path inequalities, detailed in the next subsection.

### 6.2.2 Separation problem

Given an binary vector $x$ that satisfies (6.1), (6.2), and does not contain cycles, $x$ is feasible for $\mathcal{U}^\Gamma$-$RI$ and $\mathcal{U}^\Gamma$-$\mathcal{P}I$ if and only if $x$ satisfy all constraints (6.18), or equivalently, we can define a vector $y^*$ such that $(x^*, y^*)$ satisfies all constraints (6.16). We study next the separation problem associated to these constraints. Since $x$ is binary, we know that the vector describes one path from $o$ to $d$ for each vehicle in $K$.

**Proposition 1.** *Let $p = (o = i_0, \ldots, i_{n+1} = d)$ be a path in $G$ from $o$ to $d$ for vehicle $k \in K$. The question whether $p \in \mathcal{P}^k$ can be answered in $(n - \Gamma' + 1)\Gamma'$ steps where $\Gamma' = \min\{\Gamma, n\}$.*

*Proof.* Let $\alpha(i_j)$ be the earliest arrival time at node $i_j \in p$ when the travel times are deterministic, which is formally defined by

$$\alpha(i_j) = \max\{\underline{b}_{i_j}, \alpha(i_{j-1}) + u_{i_{j-1}i_j}\}.$$

In that case, the question whether $p \in \mathcal{P}^k$ would be answered by checking that

$$\alpha(i_j) \leq \overline{b}_{i_j}, \qquad 1 \leq j \leq n,$$

which can be done in $O(n)$.

Let $\alpha(i_j, \gamma)$ be the earliest arrival time at $i_j$ when exactly $\min(j, \gamma)$ arcs are using their maximum time in subpath $i_0, \ldots, i_j$. The robust version of the earliest arrival at $i_j$ becomes the following recursive function $\alpha(i_j, \gamma) =$

$$\begin{cases} \alpha(i_0, \gamma) = \underline{b}_{i_0} & 0 \leq \gamma \leq \Gamma' \\ \alpha(i_j, 0) = \max\{\underline{b}_{i_j}, \alpha(i_{j-1}, 0) + \overline{u}_{i_{j-1}i_j}\} & 1 \leq j \leq n \\ \alpha(i_j, \gamma) = \max\{\underline{b}_{i_j}, \alpha(i_{j-1}, \gamma - 1) + \overline{u}_{i_{j-1}i_j} + \hat{u}_{i_{j-1}i_j}, \alpha(i_{j-1}, \gamma) + \overline{u}_{i_{j-1}i_j}\} & \\ & 1 \leq \gamma \leq j \\ \alpha(i_j, \gamma) = -\infty & j < \gamma \end{cases}$$

The question whether $p \in \mathcal{P}^k$ is answered by checking if

$$\alpha(i_j, \min(j, \Gamma)) \leq \overline{b}_{i_{n+1}}, \qquad 1 \leq j \leq n,$$

which can be done in $(n - \Gamma' + 1)\Gamma'$ steps. □

One observes that if $\Gamma' = \Gamma$, the separation problem for constraints (6.16) and (6.18) is solved in $O(n\Gamma)$. On the other hand, if $\Gamma' = n$, the problem is solved in $O(n)$. Notice that since each vehicle is considered independently from the others, the dynamic programming approach proposed in Proposition 1 can also be applied to the uncertainty polytope where different values of $\Gamma$ are associated with different vehicles.

Since $\mathcal{U}^\Gamma$-$\mathcal{P}I$ is solved through a branch-and-cut algorithm, it is important to be able to separate inequalities (6.18) when $x$ is fractional, which is more complicated because the

arcs on which $x^k$ takes positive values do not define a single path from $o$ to $d$. However, as explained by Kallehauge et al. [2007], we can use the fact that there is only a polynomial number of paths for which the associated path inequality (6.18) is violated. Moreover, since paths inequalities are weak, works addressing $VRPTW$ or the asymmetric traveling salesman problem with time windows rather use a lifted version of the path inequalities called tournament inequalities [Ascheuer et al., 2000].

In this work, we also separate tournament inequalities rather than paths inequalities whether $x$ is fractional or not. The only difference between our separation heuristic and the one from Kallehauge et al. [2007] is that we need to apply the dynamic programming procedure from Proposition 1 to check whether the time windows are satisfied along a candidate path. Also, in the case where $x^k$ is binary, that is, $x^k$ defines a unique path $p$ from $o$ to $d$, the tournament inequality defined for $p$ is generated as soon as $x^k$ violates the path inequality (6.18) for $p$.

### 6.2.3 Variable uncertainty

We see that whenever $x$ is binary the separation problem can be applied to $\mathcal{U}^\gamma$ exactly as in Proposition 1 since $x$ is fixed when solving the separation problem. The only subtlety concerns the choice of $\Gamma$, which is then equal to $\gamma(x)$ instead of $\Gamma$. The separation of (6.16) and (6.18) does not carry over to $\mathcal{U}^\gamma$-$RI$ and $\mathcal{U}^\gamma$-$\mathcal{P}I$ straightforwardly. For $\mathcal{U}^\gamma$-$\mathcal{P}I$, an infeasible path leads to the addition of a violated path inequality (6.18) which will forbid only the path defining the inequality. Therefore, if $\gamma^k$ only depends on $x^k$, i.e.

$$\frac{\partial \gamma^k}{\partial x^h} = 0, \text{ if } k \neq h,$$

which is a reasonable assumption in practice, the row-and-column generation and its branch-and-cut version stay valid to handle model $\mathcal{U}^\gamma$. Yet, notice that in that case, one must consider the original (weak) inequality (6.18) because only the infeasible path can be forbidden. We cannot use the aforementioned tournament inequalities because they may cut out shorter paths that would be feasible in the context of variable uncertainty.

This relatively nice picture does not extend to $\mathcal{U}^\gamma$-$RI$. In that case, the system of variables and constraints (6.17) added to $\mathcal{U}^*$-$RI$ may forbid other paths than those used in the separation problem from Proposition 1, so that they are not valid for $\mathcal{U}^\gamma$-$RI$. For the same reason, variable uncertainty cannot be combined with other types of time windows considered in this chapter.

### 6.2.4 Numerical results

Our numerical results are realized on realistic instances created for maritime ship routing and scheduling problems. In particular, the instance generator specifies the possible delay in sailing time for each arc in the network. This delay is calculated based on the time

normally required to perform the transportation represented by the arc. The delay also depends on the specific pickup port and delivery port involved, where some ports are associated with more delay than others. Such a structure is reasonable since bad weather affects the schedule more severely in some areas. Since the planning horizon is long, there is a significant risk of a ship being delayed at some point during its route, but the probability of experiencing longer travel times for all legs would be small. Hence it makes sense to make routes that can handle some delays, with $\Gamma$ equal to some small number.

The computational testing contains instances with 30, 40, and 50 different cargoes (value of $|N|$). For each number of cargoes, we consider four values of $\Gamma$: 0 (deterministic case), 1 (low uncertainty), $3 + (|N| - 20)/10$ (middle uncertainty), and $5 + (|N| - 20)/5$ (high uncertainty). For each number of cargoes, we also consider three number of ships: 1, $3 + (|N| - 20)/10$, and $5 + (|N| - 20)/5$. Finally, we generate five instances for each combination of values for the number of cargoes and number of ships.

All models and algorithms have been coded using the modeling language Xpress Mosel 3.2.3 and solved by Xpress Optimizer 22.01.09 [FICO, 2011]. A time limit of 1800 seconds has been set for each instance. They were run on a computer equipped with a processor Intel Core i5 at 2.53 GHz and 4 GB of RAM memory. The objectives of this section are (i) assessing the computational cost of solving the robust models as compared to their deterministic counterparts, and (ii) comparing formulations $\mathcal{U}^{\Gamma}$-$RI$ and $\mathcal{U}^{\Gamma}$-$\mathcal{P}I$.

Table 6.2 reports the average numbers of cuts generated by $\mathcal{U}^{\Gamma}$-$\mathcal{P}I$ for each number of cargoes and uncertainty level. We see that $\mathcal{U}^{\Gamma}$-$\mathcal{P}I$ generates significantly more cuts than $\mathcal{U}^{\Gamma}$-$RI$ generates extreme points. This can be explained by the fact that the cuts generated by $\mathcal{U}^{\Gamma}$-$\mathcal{P}I$ are tournament inequalities whose violation is checked at many nodes in the branch-and-cut tree solving $\mathcal{U}^{\Gamma}$-$\mathcal{P}I$. In opposition to this, extreme points are generated for $\mathcal{U}^{\Gamma}$-$RI$ only after an optimal integer solution has been found for the previous set of extreme points.

| | Uncertainty level ($\Gamma$) | | |
|---|---|---|---|
| $\|N\|$ | low | mid | high |
| 30 | 3.1 | 9.33 | 9.13 |
| 40 | 6.67 | 20.7 | 21.7 |
| 50 | 7.93 | 22.8 | 23.2 |

Table 6.1: Average numbers of extreme points generated by $\mathcal{U}^{\Gamma}$-$RI$.

| | Uncertainty level ($\Gamma$) | | | |
|---|---|---|---|---|
| $\|N\|$ | det | low | mid | high |
| 30 | 1210 | 313 | 867 | 795 |
| 40 | 25097 | 9501 | 17997 | 17870 |
| 50 | 17919 | 10364 | 24547 | 25072 |

Table 6.2: Average numbers of cuts generated by $\mathcal{U}^{\Gamma}$-$\mathcal{P}I$.

Average solution times are presented in Table 6.3 for each group of 5 instances. Rows entitled "av" compute the average of the three rows above them. The table presents average solution times for the two approaches. The numbers of unsolved instances within the 1800 seconds are given in parentheses and their values have been set to 1800 seconds when computing the averages. We see from Table 6.3 that the performance of $\mathcal{U}^{\Gamma}$-$RI$

and $\mathcal{U}^\Gamma$-$\mathcal{P}I$ are comparable, although $\mathcal{U}^\Gamma$-$RI$ seems to be more efficient for the larger instances. The results for both approaches present, however, important differences. The solution times for $\mathcal{U}^\Gamma$-$RI$ are highly impacted by the value of $\Gamma$. Deterministic instances are always solved faster than robust instances. Moreover, the number of extreme points of the uncertainty sets also influences the solution times since instances with uncertainty sets defined by few extreme points (low) are solved faster than instances with uncertainty sets defined by larger number of extreme points (mid and high). In opposition to this, the presence of uncertainty does not seem to influence much the solution times of $\mathcal{U}^\Gamma$-$\mathcal{P}I$.

| | | $\mathcal{U}^\Gamma$-$RI$ | | | | $\mathcal{U}^\Gamma$-$\mathcal{P}I$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| | $\Gamma$ | det | low | mid | high | det | low | mid | high |
| $\lvert N\rvert$ | $\lvert K\rvert$ | | | | | | | | |
| | 1 | 0.1 | 0.1 | 0.1 | 0.1 | 1 | 1.2 | 4.8 | 2.9 |
| 30 | 4 | 0.6 | 1.1 | 5.3 | 4.1 | 2.9 | 1.6 | 1.7 | 2.1 |
| | 7 | 3.3 | 8.2 | 58.1 | 66.1 | 27.3 | 7.6 | 16.6 | 17 |
| | av | 1.6 | 3.3 | 21.2 | 23.4 | 10.4 | 3.5 | 7.7 | 7.3 |
| | 1 | 0.1 | 0.1 | 0.3 | 1.4 | 1.5 | 2.2 | 5.1 | 24 |
| 40 | 5 | 11.3 | 160 | 640 (2) | 617 (2) | 391 (1) | 30 | 249 | 227 |
| | 9 | 364 (1) | 368 (1) | 477 (1) | 444 (1) | 452 (1) | 391 (1) | 429 (1) | 427 (1) |
| | av | 125 | 176 | 372 | 354 | 281 | 141 | 228 | 226 |
| | 1 | 0.1 | 0.3 | 1.2 | 2.8 | 6.7 | 7.3 | 28.2 | 55.3 |
| 50 | 6 | 13.8 | 31.1 | 537 (1) | 485 (1) | 534 (1) | 216 | 805 (2) | 779 (2) |
| | 11 | 109 | 748 (2) | 1070 (3) | 962 (3) | 1020 (3) | 773 (2) | 1160 (3) | 1150 (3) |
| | av | 41 | 260 | 536 | 483 | 520 | 332 | 664 | 661 |

Table 6.3: Average solution times in seconds for $\mathcal{U}^\Gamma$-$RI$ and $\mathcal{U}^\Gamma$-$\mathcal{P}I$ for larger instances.

## 6.3 Left and right soft

The constraints that couple $x$ and $y$ in the previous sections are

$$y_i^u - y_j^u + \sum_{k \in K:(i,j)\in A^k} (M + u_{ij}^k) x_{ij}^k \leq M, \quad (i,j) \in A, u \in \mathcal{U}_{diag}^\Gamma. \tag{6.19}$$

These constraints are not sufficient in the case of soft left time windows, because the arrival times (represented by $y$) would try to increase their values artificially to avoid the earliness costs. Hence, we must also consider the reverse set of inequalities in this section, which reads

$$y_i^u - y_j^u + \sum_{k \in K:(i,j)\in A^k} (-M + u_{ij}^k) x_{ij}^k \geq -M, \quad (i,j) \in A, u \in \mathcal{U}_{diag}^\Gamma. \tag{6.20}$$

Notice then that the arrival time at any node $i$ can be as large as the value of the longest path from $o$ to $i$ using the worst vector of traveling times $u \in \mathcal{U}^\Gamma$. Hence, computing

minimal values for each $M$ amounts to solve $\mathcal{NP}$-hard optimization problems, so that heuristics are required in practice. For the sake of simplicity, we simply keep $M$ in the following formulation

$$\begin{array}{ll} & \min\Big\{ \sum_{k\in K}\sum_{(i,j)\in A^k} \kappa_{ij}^k x_{ij}^k + \max_{u\in\mathcal{U}_{diag}^\Gamma} \sum_{i\in N}\sum_{k\in K} \varphi_i^{ku} \ : \\ \mathcal{U}^\Gamma\text{-}LSRS & \quad (6.1), (6.2), (6.11), (6.19), (6.20), y\geq 0, x \text{ binary}, \\ & \quad \varphi_i^{ku} \geq \max(s_i^k(\underline{b}_i - y_i^u), 0, p_i^k(y_i^u - \overline{b}_i)), \quad i\in N, k\in K, u\in\mathcal{U}_{diag}^\Gamma\Big\}, \end{array}$$

where the second term of the objective function can be replaced by a new variable $\theta$ and adding the following robust constraint to the problem

$$\theta \geq \sum_{i\in N}\sum_{k\in K} \varphi_i^{ku}, \quad u\in\mathcal{U}_{diag}^\Gamma.$$

Similarly to the situation ecountered in the previous section, $\mathcal{U}^\Gamma$-$LSRS$ can be used as such only for very small instances. To handle larger problems, one should rely instead on the the relaxed master problem $\mathcal{U}^*$-$LSRS$ defined for a subset $\mathcal{U}^* \subset \prod_{k\in K}\mathcal{U}^\Gamma$

$$\begin{aligned} \min\Big\{ & \sum_{k\in K}\sum_{(i,j)\in A^k} \kappa_{ij}^k x_{ij}^k + \max_{u\in\mathcal{U}_{diag}^\Gamma} \sum_{i\in N}\sum_{k\in K} \varphi_i^{ku} \ : \\ & (6.1), (6.2), y\geq 0, x \text{ binary}, y_o^u = 0, u\in\mathcal{U}^*, \\ & y_i^u - y_j^u + \sum_{k\in K:(i,j)\in A^k} (M + u_{ij}^k) x_{ij}^k \leq M, \quad (i,j)\in A, u\in\mathcal{U}^*, \qquad (6.21) \\ & y_i^u - y_j^u + \sum_{k\in K:(i,j)\in A^k} (-M + u_{ij}^k) x_{ij}^k \geq -M, \quad (i,j)\in A, u\in\mathcal{U}^*, \qquad (6.22) \\ & \varphi_i^{ku} \geq \max(s_i^k(\underline{b}_i - y_i^u), 0, p_i^k(y_i^u - \overline{b}_i)), \quad i\in N, k\in K, u\in\mathcal{U}^*\Big\}. \qquad (6.23) \end{aligned}$$

Then, solution algorithms iterate between solving $\mathcal{U}^*$-$LSRS$ and solving separation problems defined below, where set $\mathcal{U}^*$ is incremented with the vector $u^*\in\prod_{k\in K}\mathcal{U}^\Gamma$ found in the separation problems.

We finish the section by detailing the separation problems. Let $x^*$ be a binary vector feasible for $\mathcal{U}^*$-$LSRS$. Thus, $x^*$ characterizes $|K|$ paths from $o$ to $d$, denoted $\pi^1,\ldots,\pi^{|K|}$, which partition the nodes of $N^*$. Let $|\pi^k|$ denote the number of arcs in $\pi^k$, and define $n(\pi^k,\ell)$ as the $\ell$-th node in the path, for $\ell = 1,\ldots,|\pi^k|+1$. In particular, $n(\pi^k,1) = o$ and $n(\pi^k,|\pi^k|+1) = d$. We define similarly $a(\pi^k,\ell)$ as the $\ell$-th arc in the path. With these definitions, we see that

$$y_{n(\pi^k,\ell)}^u = \sum_{t=1}^{\ell-1} u_{a(\pi^k,t)} \qquad (6.24)$$

for any $u \in \prod_{k\in K} \mathcal{U}^\Gamma$ and $\ell = 2, \ldots, |\pi^k|$. Therefore, one readily verifies that, for each $k \in K$, the separation problem can be written

$$\max_{u\in\mathcal{U}^\Gamma} g^k(u) = \max_{u\in\mathcal{U}^\Gamma} \sum_{\ell=2}^{|\pi^k|} \max\left\{ s^k_{n(\pi^k,\ell)} \left( \underline{b}_{n(\pi^k,\ell)} - \sum_{t=1}^{\ell-1} u_{a(\pi^k,t)} \right), 0, \right.$$
$$\left. p^k_{n(\pi^k,\ell)} \left( \sum_{t=1}^{\ell-1} u_{a(\pi^k,t)} - \overline{b}_{n(\pi^k,\ell)} \right) \right\}.$$

Notice that the outer summation starts at $\ell = 2$ because there is no time window associated to the depot $o = n(\pi^k, 1)$. Since each path $\pi^k$ is fixed when solving the separation problem, let us consider a specific $k$ and reorder the nodes and the arcs of the graph as

$$(n(\pi^k, 2), n(\pi^k, 3), \ldots, n(\pi^k, |\pi^k|), \text{ other nodes in arbitrary order}), \tag{6.25}$$

and

$$(a(\pi^k, 1), a(\pi^k, 2), \ldots, a(\pi^k, |p^k| - 1), \text{ other arcs in arbitrary order}), \tag{6.26}$$

respectively. With these reorderings, $g^k$ can be written as

$$g^k(u) = \max_{u\in\mathcal{U}^\Gamma} \sum_{\ell=1}^{|\pi^k|-1} \max\left\{ s^k_\ell \left( \underline{b}_\ell - \sum_{t=1}^{\ell} u_t \right), 0, p^k_\ell \left( \sum_{t=1}^{\ell} u_t - \overline{b}_\ell \right) \right\},$$

and we easily see the similarity of $g^k$ with the function $g$ defined in (5.7) from the previous chapter. It is therefore not surprising that a very similar dynamic programming algorithm can be used to solve $\max_{u\in\mathcal{U}^\Gamma} g^k(u)$ in pseudo-polynomial time. Altough the details of the algorithm are omitted for brevity, it is very close to the one described in Section 5.3. Finally, we obtain the new vector $u^* \in \prod_{k\in K} \mathcal{U}^\Gamma$ by concatenating the vectors $u^{1*}, \ldots, u^{|K|*}$ obtained for the $|K|$ separation problems.

## 6.4 Left soft and right hard

The situation with right hard time windows is very similar to the case studied in the previous section. We jump immediately to the definition of the relaxed master problem $\mathcal{U}^*$-$LSRH$, defined for a subset $\mathcal{U}^* \subset \prod_{k\in K} \mathcal{U}^\Gamma$

$$\min \Big\{ \sum_{k\in K} \sum_{(i,j)\in A^k} \kappa^k_{ij} x^k_{ij} + \max_{u\in\mathcal{U}^\Gamma_{diag}} \sum_{i\in N} \sum_{k\in K} \varphi^{ku}_i \ :$$
$$(6.1), (6.2), (6.21), (6.22), y \geq 0, x \text{ binary}, y^u_o = 0, u \in \mathcal{U}^*,$$
$$\varphi^{ku}_i \geq \max(s^k_i(\underline{b}_i - y^u_i), 0), \quad i \in N, k \in K, u \in \mathcal{U}^* \tag{6.27}$$
$$y^u_i \leq \overline{b}_i, \qquad i \in N^*, u \in \mathcal{U}^* \quad \Big\}. \tag{6.28}$$

There are two differences between $\mathcal{U}^*$-$LSRH$ and $\mathcal{U}^*$-$LSRS$. First, the maximization in (6.27) contains only two terms (unlike the one from (6.23) that contains three terms), thus making the resulting separation problem even more similar to the one studied for the lot-sizing problem in Section 5.3. Second, we also have to ensure that constraints (6.28) are satisfied. Using equation (6.24), one readily verifies that the separation problem associated to (6.28) comes down to solving $\max_{u\in\mathcal{U}^\Gamma} \sum_{t=1}^{\ell-1} u_{a(\pi^k,t)}$, which can be done in polynomial time through a sorting algorithm.

## 6.5 Left hard and right soft

We shall see in this section that the presence of left hard time windows hardens the situation observed so far. As before, let us define the relaxed master problem as

$$\begin{aligned}
&\min\Big\{ \sum_{k\in K}\sum_{(i,j)\in A^k} \kappa_{ij}^k x_{ij}^k + \max_{u\in\mathcal{U}^\Gamma_{diag}} \sum_{i\in N}\sum_{k\in K} \varphi_i^{ku} \ : \\
\mathcal{U}^*\text{-}LHRS \qquad & (6.1),(6.2),(6.21),(6.22), y\geq 0, x \text{ binary}, y_o^u = 0, u\in\mathcal{U}^*, \\
& \varphi_i^{ku} \geq \max(0, p_i^k(y_i^u - \bar{b}_i)), \quad i\in N, k\in K, u\in\mathcal{U}^* \\
& \underline{b}_i \leq y_i^u, \qquad i\in N^*, u\in\mathcal{U}^* \quad \Big\}. \qquad (6.29)
\end{aligned}$$

In the rest of the section, we focus on a given vehicle $k$ and on the route $\pi^k$ described by an integer solution $x^*$ to $\mathcal{U}^*$-$LHRS$. In particular, we reorder the nodes and the arcs as explained in (6.25) and (6.26). Constraints (6.29) break the structure witnessed for $\mathcal{U}^*$-$LSRH$ and $\mathcal{U}^*$-$LSRS$ because the arrival time $y_i^u$ at node $i$ can no longer be defined as an affine function of $u$, such as (6.24). Namely, the wait of the vehicle in case it arrives at node $i$ before $\underline{b}_i$ yields arrival times that can be formulated recursively as

$$y_i^u = \max(\underline{b}_i, y_{i-1}^u + u_{i-1}), \qquad (6.30)$$

for each $i = 2,\ldots,|\pi^k|$. Therefore, it is not possible to adapt the DPA from Section 5.3 to maximize penalty function

$$f(u) = \sum_{i=1}^{|\pi^k|-1} \max\{s_i^k(y_i^u - \bar{b}_i), 0\},$$

with $y_i^u$ defined in (6.30).

Fortunately, it is possible to adapt our $DPA$ to handle this difficulty. Let us denote the separation problem as

$$AP \qquad \max_{u\in\mathcal{U}^\Gamma} f(u),$$

and denote $f_i(u) = \max\{s_i^k(y_i^u - \overline{b}_i), 0\}$. Expanding recursively the maxima in (6.30), $y_i^u$ can be expressed as (see Agra et al. [2012], Chardy and Klopfenstein [2010])

$$y_i^u = \max_{\ell=1,\dots,i} \left\{ \underline{b}_\ell + \sum_{t=\ell}^{i} u_t \right\}. \tag{6.31}$$

We can use (6.31) to rewrite each term $f_i$ of the penalty function as

$$\begin{aligned} f_i(u) &= s_i^k \left[ \max_{\ell=1,\dots,i} \left\{ \underline{b}_\ell + \sum_{t=\ell}^{i} u_t \right\} - \overline{b}_i \right]^+ \\ &= \max_{\ell=1,\dots,i} \left\{ s_i^k \left[ \underline{b}_\ell + \sum_{t=\ell}^{i} u_t - \overline{b}_i \right]^+ \right\}, \end{aligned}$$

where we used the simplified notation $[x]^+$ for $\max\{0, x\}$.

Let $u_{[\ell i]}$ denote the subvector $\{u_t : t = \ell, \dots, i\}$ and define $\overline{f}_{\ell i}(u_{[\ell i]}) = s_i^k[\underline{b}_\ell - \overline{b}_i + \sum_{t=\ell}^{i} u_t]^+$. Hence, $f_i(u) = \max_{\ell=1,\dots,i} \overline{f}_{\ell i}(u_{[\ell i]})$. Let $\beta(m, \gamma)$ be the value of the optimal solution of the restricted problem defined for the subpath $1, \dots, m$ with at most $\gamma$ deviations:

$$\beta(m, \gamma) = \max_{u \in \mathcal{U}^\gamma_{[1m]}} \left\{ \sum_{i=1}^{m} \max_{\ell=1,\dots,i} \overline{f}_{\ell i}(u_{[\ell i]}) \right\},$$

where

$$\mathcal{U}^\gamma_{[\ell i]} := \left\{ u : u_t = \overline{u}_t + \hat{u}_t \xi_t, t = \ell, \dots, i, \xi \in \{0,1\}^{i-\ell+1}, \sum_{t=\ell}^{i} \xi_t \le \gamma \right\}.$$

Clearly, $\text{opt}(AP) = \beta(n, \Gamma)$.

The rest of the section is devoted to the construction of a DPA to compute $\beta(n, \Gamma)$. Notice that for any $t$, $\sum_{i=t}^{m} \overline{f}_{ti}$ has the structure of the function studied in Section 5.3, so that the sum can be optimized over the set $\mathcal{U}^\gamma_{[tm]}$ in pseudo-polynomial time by applying the DPA presented therein. Let us denote $\overline{f}^\beta(u_{[1m]}) = \sum_{i=1}^{m} \max_{\ell=1,\dots,i} \overline{f}_{\ell i}(u_{[\ell i]})$ so that $\beta(m, \gamma)$ can be rewritten as

$$\beta(m, \gamma) = \max_{u \in \mathcal{U}^\gamma_{[1m]}} \overline{f}^\beta(u_{[1m]}).$$

The DPA from Section 5.3 cannot be used directly to optimize $\overline{f}^\beta(u_{[1m]})$ because of the maximization involved in the definition of $\overline{f}^\beta(u_{[1m]})$. Hence, we use next an alternative recursion based on the key lemma below. The lemma expresses $\overline{f}^\beta(u_{[1m]})$ from the set of functions $\left\{ \overline{f}^\beta(u_{[1t]}) : 1 \le t \le m-1 \right\}$ and the sums $\left\{ \sum_{i=t}^{m} \overline{f}_{ti}(u_{[ti]}) : 1 \le t \le m \right\}$. We show in the balance of the section how this leads to a DPA.

**Lemma 14.** *Let $u \in \mathcal{U}^\gamma$ be fixed and $m \in \{2\dots,n\}$. It holds that*

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

*Proof.* Since $u$ is fixed, we simplify notations and denote $\overline{f}_{\ell i}(u_{[\ell i]})$ 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 $i \in \{\ell,\dots,m\}$:

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

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

$$\overline{f}_{tm} = \max_{\ell=1,\dots,m} \overline{f}_{\ell m} \quad \Rightarrow \quad \overline{f}_{ti} = \max_{\ell=1,\dots,i} \overline{f}_{\ell i} \text{ for all } i = t,\dots,m. \tag{6.32}$$

The counterpart of (6.32) for the whole sum $\sum_{i=1}^{m} \max_{\ell=1,\dots,i} \overline{f}_{\ell i}$ is

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

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

Using Lemma 14 we have, for each $2 \le m \le n$ and $0 \le \gamma \le \Gamma$:

$$\begin{aligned}
\beta(m,\gamma) &= \max_{u \in \mathcal{U}^\gamma_{[1m]}} \left\{ \overline{f}^\beta(u_{[1m]}) \right\} \\
&= \max_{u \in \mathcal{U}^\gamma_{[1m]}} \max_{t=1,\dots,m} \left\{ \overline{f}^\beta(u_{[1(t-1)]}) + \sum_{i=t}^{m} \overline{f}_{ti}(u_{[ti]}) \right\} \\
&= \max_{t=1,\dots,m} \max_{\delta=0,\dots,\gamma} \left\{ \max_{u \in \mathcal{U}^\delta_{[1(t-1)]}} \overline{f}^\beta(u_{[1(t-1)]}) + \max_{u \in \mathcal{U}^{\gamma-\delta}_{[tm]}} \sum_{i=t}^{m} \overline{f}_{ti}(u_{[ti]}) \right\} \\
&= \max_{t=1,\dots,m} \max_{\delta=0,\dots,\gamma} \left\{ \beta(t-1,\delta) + \max_{u \in \mathcal{U}^{\gamma-\delta}_{[tm]}} \sum_{i=t}^{m} \overline{f}_{ti}(u_{[ti]}) \right\} \\
&= \max_{t=1,\dots,m} \max_{\delta=0,\dots,\gamma} \left\{ \beta(t-1,\delta) + \overline{F}(t,m,\gamma-\delta) \right\},
\end{aligned} \tag{6.33}$$

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

$$\beta(1,\gamma) = \max_{u \in \mathcal{U}^\gamma_{[11]}} \overline{f}_{11}(u_{[11]}) = s_1[\underline{b}_1 - \overline{b}_1]^+ = 0. \tag{6.34}$$

Combining (6.33) and (6.34), we obtain a DPA to solve $AP$.

We conclude the section by showing that (6.33) yields a pseudo-polynomial DPA. Notice that function

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

is again a special case of the function studied in the previous chapter for each $1 \leq t \leq m \leq n$. Hence, we can apply the DPA from Section 5.3 to compute $\overline{F}(t, m, \gamma) = \max_{u \in \mathcal{U}^{\gamma}_{[tm]}} \bar{f}_{tm}(u_{[tm]})$ for each $1 \leq t \leq m \leq n$ and $0 \leq \gamma \leq \Gamma$. Namely, let $\alpha_t$ be the table used to compute $\overline{F}(t, n, \Gamma)$ through the DPA from Section 5.3. 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 5.3 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, (6.33) can be solved in $O(n^2\Gamma^2)$. Since $\Gamma \leq \overline{\phi}$, we obtain the following worst-case complexity.

**Lemma 15.** *Problem AP can be solved by a DPA in $\mathcal{O}(n^2\Gamma\overline{\phi})$ operations.*

# Chapter 7

# Conclusion and future work

The objectives of the thesis were both numerical and theoretical. From the numerical viewpoint, we have proposed decomposition algorithms that split the optimization problems into two parts. On the one side, the relaxed master problem contains the combinatorial structure of the problem and the robust constraints associated with small subsets of the uncertainty sets. On the other side, the true robust constraints are handled by separation problems, that are solved efficiently by exploiting the combinatorial structure of $\mathcal{U}^\Gamma$. From the theoretical viewpoint, we have answered our original question regarding easy problems turning $\mathcal{NP}$-hard, and realized on the way that achievable approximation ratios do not generalize easily to the robust context. Generally speaking, our opinion is that the field of robust combinatorial optimization with budgeted uncertainty is everything but covered, and we list in the next sections some research directions that we intend to pursue in the following years.

## 7.1 Numerical research directions

**Variable uncertainty** The new model presented in Chapter 2 would need further validation to prove useful in practice. Specifically, one should consider applications with real data and see how using $\mathcal{U}^\gamma$ yields cheaper solutions than $\mathcal{U}^\Gamma$, while guaranteeing similar reliability levels. This would require problems where one wishes that individual probabilistic constraints be satisfied while having little information over the true probability distributions.

**Efficient implementation for the constrained shortest path problem** It is known that the label-setting algorithm can be sped up significantly for capacity constraints through reduction rules and bounds obtained from the Lagrangian relaxation of the capacity constraint. We are currently working with Luigi di Puglia Pugliese and Francesca Guerriero (Unversity of Calabria) on generalizing those for the robust constrained shortest path problem studied in Chapter 3, hopefully providing efficient algorithms for $\mathcal{U}^\Gamma$-$CSP$.

**MIP with pseudo-polynomial separation algorithms** The results of Chapters 5 and 6 could be improved along the following three directions. First, the exact pseudo-polynomial dynamic programming algorithms should be called only when heuristic solutions are unable to find violated extreme points, in the line of well-engineered branch-and-cut algorithms commonly developed in the mixed-integer programming literature. Second, the dynamic programming algorithms should be improved to incorporate dominance rules and other state space reduction techniques. Third, integer programs such as the vehicle routing problems studied in Chapter 6 should be addressed by branch-and-cut-and-price algorithms rather than row-and-column generation algorithms. This would avoid the exploration of several branch-and-bound trees. The difficulty with the third aspect is that coding efficient branch-and-cut-and-price algorithms is not easy, as commercial solvers do not handle the dynamic generation of variables along the branch-and-bound tree. One should instead rely on the existing academic platforms such as SCIP or BaPCoD. We have recently started to explore this third direction with the platform BaPCoD, developed by the team-project Realopt from Bordeaux. The research is carried out in collaboration with Zacharie Ales from the University of Avignon, and Boris Detienne and François Vanderbeck from the University of Bordeaux.

**Paths formulations for routing problems** The most competitive algorithms for vehicle routing problems are based on advanced branch-and-cut-and-price algorithms where routes, rather than scenarios, are priced out along the search tree. It would therefore be natural to see how the label-setting algorithms developed in Chapter 3 can be combined with the advanced algorithmic techniques that have been proposed for vehicle routing problems, see Pecin et al. [2014], to improve the results provided in Chapter 6. We intend to work on the topic with Artur Alves Pessoa (Federal Fluminense University) in the future.

## 7.2 Theoretical research directions

**Variable uncertainty** One can easily see that the concept of variable uncertainty can be applied to any point-to-set mapping, although the resulting optimization problems may be harder to solve than their classical robust counterparts. It would be interesting to find other types of point-to-set mappings that are useful from a theoretical viewpoint. An idea would be to combine the distributionally robust optimization paradigm with variable uncertainty.

**Scheduling** We think that the main theoretical contribution of this work has been to start the investigation of robust scheduling, which is a virtually empty field of research. Much is left to do on the topic and we are currently working on extensions of our results

in many directions. Immediate extensions of the work presented in the thesis concern the following questions.

1. Dualizing the assignment formulation for $1||\mathcal{U}_p^\Gamma|\sum_j w_jC_j$, one ends up with a problem combining $1||\sum_j w_jT_j$ and $1||\sum_j w_jC_j$. Can this be exploited to provide pseudo-polynomial time algorithms for $1||\mathcal{U}_p^\Gamma|\sum_j w_jC_j$ ?

2. Provide positive and/or negative results on the approximation ratios achievable for $1||\mathcal{U}_p^\Gamma|\sum_j w_jC_j$.

3. Provide constant factor approximation algorithms for $Q||\mathcal{U}_p^\Gamma|C_{max}$ and $R||\mathcal{U}_p^\Gamma|C_{max}$.

4. When $\Gamma$ is fixed, we could provide a PTAS for $P||\mathcal{U}_p^\Gamma|C_{max}$, which has been left out of this manuscript for brevity. Is the problem still PTAS when $\Gamma$ is part of the input? The first step in that direction would be to find an $(1+\epsilon)$-approximation ratio whenever all jobs have $\overline{u}_j$ and $\hat{u}_j$ less than $\epsilon$.

In addition to these immediate extensions, we would like to get a more profound understanding of the type of deterministic algorithms that can be generalized to the robust context, to avoid having to study each of them individually. We are currently working on these research directions together with Marin Bougeret (University of Montpellier) and Artur Alves Pessoa.

**Lot-sizing** It is unfortunate that we were not able to prove that the robust lot-sizing problem is $\mathcal{NP}$-hard. While very special cases have been proved polynomial (constant values of $\Gamma$ and $\Gamma = n$), the complexity of the general problem is still unknown.

## 7.3 Lot-sizing

In addition to the above well-identified research directions, our preliminary results on robust lot-sizing problems lead us to think that these could also benefit from exploiting the structure of $\mathcal{U}^\Gamma$. However, general robust lot-sizing problems should be addressed in a multi-stage way, instead of the static approach used in Chapter 5. Namely, the production vector $x$ should depend on past realizations of the demand, which is modeled through the non-anticipativity constraints. Multi-stage robust optimization problems are very challenging both from the numerical and theoretical viewpoints since the non-anticipativity constraints prevent us from applying the Benders-like decomposition algorithms used in this thesis. Relaxing these constraints make the problem tractable numerically in some cases [Santos et al., 2016b], but the resulting solutions provide only lower bounds for the original problem.

Alternatively, Minoux [2009] proposes simple dynamic programming algorithms able to handle a specific class of robust-lot-sizing problems, without suffering the curse of dimensionality often encountered in multi-stage problems. Can this algorithm be generalized to more general problems?

## 7.4 Beyond budgeted uncertainty

A different type of interesting open questions is related to the generalization of Theorems 2 and 3 to more general *structured* uncertainty sets. We are currently working on a generalization of these theorems to uncertainty sets that correspond to the extreme points of polytopes defined by small numbers of knapsack constraints. We are also working on approximation algorithms for the robust counterparts of approximable problems $CO$ when the uncertainty set is an axis-parallel ellipsoid

$$\mathcal{U}_{ap}^{ell} := \left\{ u \in \mathbb{R}^n : u_i = \overline{u}_i + \xi_i \hat{u}_i, \sum_{i=1}^{n} \xi_i^2 \leq \Omega^2 \right\}.$$

In fact, a very interesting open question concerns the complexity of $\mathcal{U}_{ap}^{ell}$-$CO$ which is still unknown, contrasting with the $\mathcal{NP}$-hardness of $\mathcal{U}^{ell}$-$CO$ proved in Sim [2004], where the ellipsoid $\mathcal{U}^{ell}$ is defined as

$$\mathcal{U}^{ell} := \left\{ u \in \mathbb{R}^n : u_i = \overline{u}_i + \xi_i \hat{u}_i, \xi^T A \xi \leq \Omega^2 \right\}$$

for some positive semi-definite matrix $A$.

# Bibliography


A. Agra, M. Christiansen, R. Figueiredo, L. M. Hvattum, M. Poss, and C. Requejo. Layered formulation for the robust vehicle routing problem with time windows. In *ISCO*, pages 249–260, 2012.

A. Agra, M. Christiansen, R. Figueiredo, L. M. Hvattum, M. Poss, and C. Requejo. The robust vehicle routing problem with time windows. *Computers & Operations Research*, 40(3):856 – 866, 2013.

A. Agra, M. Christiansen, A. Delgado, and L. Simonetti. Hybrid heuristics for a short sea inventory routing problem. *European Journal of Operational Research*, 236(3):924–935, 2014.

H. Aissi, C. Bazgan, and D. Vanderpooten. Min–max and min–max regret versions of combinatorial optimization problems: A survey. *European journal of operational research*, 197(2):427–438, 2009.

M. A. Aloulou and F. D. Croce. Complexity of single machine scheduling problems under scenario-based uncertainty. *Operations Research Letters*, 36(3):338 – 342, 2008.

E. Álvarez-Miranda, I. Ljubić, and P. Toth. A note on the bertsimas & sim algorithm for robust combinatorial optimization problems. *4OR*, 11(4):349–360, 2013.

N. Ascheuer, M. Fischetti, and M. Grötschel. A polyhedral study of the asymmetric traveling salesman problem with time windows. *Networks*, 36(2):69–79, 2000.

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

A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

A. Ben-Tal, L. E. Ghaoui, and A. Nemirovski. *Robust optimization*. Princeton University Press, 2009.

D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Math. Program.*, 98(1-3):49–71, 2003.

D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52:35–53, 2004.

D. Bertsimas, D. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53:464–501, 2011.

D. Bertsimas, I. Dunning, and M. Lubin. Reformulation versus cutting-planes for robust optimization. *Computational Management Science*, pages 1–23, 2015.

D. Bienstock and N. Özbay. Computing robust basestock levels. *Discrete Optimization*, 5(2):389–414, 2008.

N. Boland, J. Dethridge, and I. Dumitrescu. Accelerated label setting algorithms for the elementary resource constrained shortest path problem. *Operations Research Letters*, 34(1):58 – 68, 2006.

Q. Botton, B. Fortz, L. Gouveia, and M. Poss. Benders decomposition for the hop-constrained survivable network design problem. *INFORMS Journal on Computing*, 25 (1):13–26, 2013.

M. Bougeret, A. A. Pessoa, and M. Poss. Robust scheduling with budgeted uncertainty, 2016. Submitted.

C. Büsing, A. M. C. A. Koster, and M. Kutschka. Recoverable robust knapsacks: Γ-scenarios. pages 583–588, 2011.

C. Büsing, F. D'Andreagiovanni, and A. Raymond. Robust optimization under multi-band uncertainty. In *12th Cologne-Twente Workshop on Graphs and Combinatorial Optimization, Enschede, Netherlands, May 21-23, 2013.*, pages 35–38, 2013.

M. Chardy and O. Klopfenstein. Handling uncertainties in vehicle routing problems through data preprocessing. In *proceedings of the Third International Conference on Information Systems, Logistics and Supply Chain*, 2010.

G. Claßen, A. M. C. A. Koster, and A. Schmeink. The multi-band robust knapsack problem - A dynamic programming approach. *Discrete Optimization*, 18:123–149, 2015.

CPLEX. *IBM ILOG CPLEX 12.5 Reference Manual*, 2013.

R. L. Daniels and P. Kouvelis. Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science*, 41(2):pp. 363–376, 1995.

M. Desrochers and F. Soumis. A generalized permanent labelling algorithm for the shortest path problem with time windows. *INFOR*, 26:191–212, 1988.

I. Dumitrescu and N. Boland. Improved preprocessing, labeling and scaling algorithms for the weight-constrained shortest path problem. *Networks*, 42:135–153, 2003.

L. El Ghaoui, F. Oustry, and H. Lebret. Robust solutions to uncertain semidefinite programs. *SIAM Journal on Optimization*, 9(1):33–52, 1998.

FICO. Xpress-Optimizer reference manual. Technical Report Release 22.01, 2011.

M. Fischetti and M. Monaci. Cutting plane versus compact formulations for uncertain (integer) linear programs. *Mathematical Programming Computation*, 4(3):239–273, 2012.

V. Gabrel, C. Murat, and A. Thiele. Recent advances in robust optimization: An overview. *European Journal of Operational Research*, 235(3):471–483, 2014.

M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

K.-S. Goetzmann, S. Stiller, and C. Telha. Optimization over integers with robustness in cost and few constraints. In *WAOA*, pages 89–101, 2011.

R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. R. Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics*, 5:287–326, 1979.

M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, Berlin, 1988.

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

O. H. Ibarra and C. E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM (JACM)*, 22(4):463–468, 1975.

B. Kallehauge, N. Boland, and O. Madsen. Path inequalities for the vehicle routing problem with time windows. *Networks*, 49(4):273–293, 2007.

L. G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

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

P. Kouvelis and G. Yu. *Robust discrete optimization and its applications*, volume 14. Springer Science & Business Media, 2013.

J. Lenstra, D. Shmoys, and E. Tardos. Approximation algorithms for scheduling unrelated parallel machines. *Mathematical Programming*, 46(1-3), 1990.

C. Miller, A. Tucker, and R. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

M. Minoux. Solving some multistage robust decision problems with huge implicitly defined scenario trees. *Algorithmic Operations Research*, 4(1):1–18, 2009.

M. Monaci and U. Pferschy. On the robust knapsack problem. *SIAM Journal on Optimization*, 23(4):1956–1982, 2013.

M. Monaci, U. Pferschy, and P. Serafini. Exact solution of the robust knapsack problem. *Computers & OR*, 40(11):2625–2631, 2013.

D. Pecin, A. Pessoa, M. Poggi., and E. Uchoa. Improved branch-cut-and-price for capacitated vehicle routing. In *Integer Programming and Combinatorial Optimization - 17th International Conference, IPCO 2014, Bonn, Germany, June 23-25, 2014. Proceedings*, pages 393–403, 2014.

A. A. Pessoa and M. Poss. Robust network design with uncertain outsourcing cost. *INFORMS Journal on Computing*, 27(3):507–524, 2015.

A. A. Pessoa, L. D. P. Pugliese, F. Guerriero, and M. Poss. Robust constrained shortest path problems under budgeted uncertainty. *Networks*, 66(2):98–111, 2015.

M. Pioro, D. Nace, M. Poss, and Y. Fouquet. An optimization model for multicommodity flow networks with partial multiple link failures. *Operations Research*, 2016. In press.

M. Poss. Robust combinatorial optimization with variable budgeted uncertainty. *4OR*, 11(1):75–92, 2013.

M. Poss. Robust combinatorial optimization with variable cost uncertainty. *European Journal of Operational Research*, 237(3):836–845, 2014.

M. Poss and C. Raack. Affine recourse for the robust network design problem: between static and dynamic routing. *Networks*, 61(2):180–198, 2013.

M. C. Santos, A. Agra, M. Poss, and D. Nace. A dynamic programming approach for a class of robust optimization problems. *SIAM Journal on Optimization*, (3):1799–1823, 2016a.

M. C. Santos, M. Poss, and D. Nace. A perfect information lower bound for robust lot-sizing problems, 2016b. Submitted.

M. Sim. *Robust optimization*. PhD thesis, Phd. Thesis, June, 2004.

W. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66, 1956.

A. L. Soyster. Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations Research*, 21:1154–1157, 1973.

J. Yang and G. Yu. On the robust single machine scheduling problem. *Journal of Combinatorial Optimization*, 6(1):17–33, 2002.

G. Yu and J. Yang. On the robust shortest path problem. *Computers & Operations Research*, 25(6):457–468, 1998.

F. B. Zhan and C. E. Noon. Shortest path algorithms: An evaluation using real road networks. *Transportation Science*, 32(1):65–73, 1998.