Université Libre de Bruxelles

Faculté des Sciences

Département d'Informatique

Models and algorithms for network design problems

Thèse présentée en vue de l'obtention
du grade de Docteur en Sciences

par

Michael POSS

Co-promoteurs:

Bernard FORTZ
Martine LABBE
François LOUVEAUX

Février 2011

Acknowledgements

First of all, I would like to warmly thank my advisors Bernard Fortz, Martine Labbé and François Louveaux for supervising my thesis. The three of you have shown me how research can be both productive and recreational. I first came to see Martine because I was interested in the real-life applications of mathematics. Her warm welcome and sensible arguments easily convinced me to start a PhD under her supervision. Martine, I thank you for the bright insights you have given me. But beyond the mathematics, I have always been impressed by your talent to motivate and stimulate research fellows. Bernard, you were always on hand to answer any doubts I had during these past years. Thank you Bernard, for all your help and support. I am glad of the experiences we have shared at GOM and in conferences. Also, I am grateful to François for the tips he gave me on stochastic programming.

I may not have begun a PhD without the positive experience I had at France Telecom R&D, working under the supervision of Adam Ouorou. Adam, you convinced me that applied mathematics, while it requires different skills, is not less interesting than differential geometry. Thanks are due to you, to Gianluca Bontempi, to Jean-François Raskin, and to Ruediger Schultz for agreeing to be part of my PhD committee.

A big thank you also goes to Nelson Maculan who warmly welcomed me at the Federal University of Rio de Janeiro. During my stay, I had the pleasure of working with Claudia Sagastizábal whom I would like to thank for the relevant discussions about the usability of mathematical models and research when facing real-life problems.

I would like to express my gratitude to the other researchers I had the great pleasure of working with at some point during the the past four years. Thanks to Luis Gouveia of the University of Lisbon for our successful collaboration. Thanks to Christian Raack of the ZIB for the effective collaboration, and for not letting me die alone on my snowboard. Thanks also to Quentin Botton of the University of Louvain for our enjoyable collaboration and discussions.

I thank my colleagues from the Computer Science Department at the ULB for their friendship, their support and the interest they took in my work. I would also like to thank the colleagues I had the chance to work with in Aveiro and Rio de Janeiro. The good atmospheres allowed me to work in very pleasant environments.

I take this opportunity to thank my friends and relatives who helped to make these years such a nice experience. I am grateful that you have always been there whenever I needed you. In particular, I would like to thank my mother, Anca, and my father, Daniel, for having always supported my choices. Special thanks are also due to Alice, who did her best to correct my “frenchy” English.

The beauty of life is that you never know what is waiting for you. Thank you, Rosa, for all the moments of happiness you have given me. Thanks to you, my PhD has taken me to countries I had only dreamed of going to. You have made my experience about so much more than just academic research.

List of publications

International Journals

[1] B. Fortz and M. P. Easy distributions for combinatorial optimization problems with probabilistic constraints, *Oper. Res. Lett.*, 38:545–549, 2010.

[2] S. L. Moulin, M. P., and C. Sagastizábal. Transmission expansion planning with re-design, *Energy Systems*, 1(2):113–139, 2010.

[3] B. Fortz and M. P. An improved benders decomposition applied to a multi-layer network design problem, *Oper. Res. Lett.*, 37:359–364, 2009.

Technical Reports

[4] Q. Botton, B. Fortz, L. Gouveia, and M. P. Benders decomposition for the hop-constrainted survivable network design problem. Technical report, GOM, Université Libre de Bruxelles, 2010, *Submitted.*

[5] B. Fortz, M. Labbé, F. V. Louveaux, and M. P. The knapsack problem with Gaussian weights. Technical Report 592, GOM, Université Libre de Bruxelles, 2009, *Submitted.*

Refereed International Conference Proceedings

[6] B. Fortz, M. Labbé and M.P. A branch-and-cut-and-price framework for convex MINLP applied to a stochastic network design problem. In the Proceedings of: *European Workshop on Mixed Integer Nonlinear Programming*, 2010.

List of Tables

Contents

1 Introduction

Our world is constantly drifting towards networking. Information, energy, goods, people and others need to be transported quickly and efficiently from one place to another. Therefore, these networks must be planned carefully, usually at the lowest cost, taking into account the specific requirements of the commodities. The network planning problem starts with the design of the network, and the choice of the link capacities. We must then decide of a routing scheme for the commodities on the network, that satisfies technical requirements, such as survivability in case of link failure, or physical laws for networks carrying fluids (gas, electricity, water, ...). Eventually, the network will need to be extended to cope with increasing demands, that is, new links will be installed or the capacity of existing ones will be increased. Herein we are more particularly interested in designing telecommunications and power transmission networks, further described in next subsection, and in extending existing ones.

Mathematical programming models optimization problems through functions and inequalities in order to find the best solution to these problems, or if no solution exists, to understand why that is so. In general, these functions can be anything from linear, to non-linear non-convex non-differentiable. Similarly, their domain of definition can be "easy" convex sets, such as polyhedra, but also complicated non-convex or even non-connected sets. Of course, the quality of the solution we are able to provide depends on the properties satisfied by these objects. In this thesis, we work more particularly in the field of Mixed-Integer Programming (MIP), meaning that some of the variables take values in discrete sets. In general, MIP requires enumeration-type algorithms called branch-and-bound, branch-and-cut, and branch-and-price algorithms.

An important subclass of MIP, called Mixed-Integer Linear Programming (MILP), considers that all functions involved in the description of the problem are linear. Even though many MILPs are still very challenging, there are now efficient and reliable tools to tackle them, even with little knowledge about their specific structure. MILP is a very general tool, allowing to model a large variety of optimization problems. However, sometimes the optimization problem presents non-linearities which must be taken into account to model the problem accurately. For instance, problems featuring uncertain parameters, such as stochastic programs, can be seen as non-linear problems. Together with integrality requirement on some of the variables, we obtain Mixed-Integer Non Linear Programming (MINLP). Herein, we study network design problems, some of them purely deterministic and others featuring uncertain parameters, within the frameworks of MILP and MINLP.

Our contributions are both practical and theoretical. Practical because we improve existing algorithms to solve close to be realistic network design models and we formulate new models yielding cheaper solutions. Theoretical because we provide complexity results for stochastic knapsack problems and general capacitated problems under specific assumptions. Our contributions are further detailed in Section 1.2 while the following section describes the two main areas of applications considered in the thesis.

1.1 Applications

In this section, we introduce the two main applications of the models studied along this thesis.

Telecommunications industry

Liberalization of telecommunication markets in Europe, which has started a few years ago, forces companies running these networks to maximize their profits (or minimize their costs). A problem which has to be dealt with is how to modify an existing network to make the best use of new technologies. Furthermore, the ever-growing demand for bandwidth creates a constant need for the planning and the expansion of telecommunication networks.

Modern telecommunication transport networks accommodate various kinds of traffic. This includes, but is not limited to, IP traffic from the Internet, traffic from dedicated virtual private networks of large companies, video traffic, and voice traffic from fixed-line or mobile telephone calls. The links of the network must thus provide sufficient capacity (reserved bandwidth) to route all this data through the network. Usually, this capacity can only be installed in discrete steps, that we call batches of capacity.

The network operator who must build or adapt an existing network to changing demands faces a network design problem. These problems can have a wide range of characteristics depending on the specific technologies and side constraints. Stated in its general form, the network design problem consists of choosing a set of links (yielding the network topology), installing capacities on the links, and routing the traffic demands of all customers within these capacities. In order to protect the traffic against link failures, spare capacity may have to be installed in the network in such a way that affected traffic can be rerouted around the failing node or link.

In this thesis, we shall consider the following two telecommunications network design problems, denoted by (**ML**) and (**HOP**). First, we consider a bi-layer network design problem, (**ML**). Given a set of point-to-point demands and two graphs that share the same node set, (**ML**) consists of choosing links in both graphs and installing capacities on these links. One of the graph represents the upper layer, while the other represents the lower layer. Enough capacity should be installed in the upper layer to route each demand in that layer. Then, capacities installed in the upper layer yield the set of the point-to-point demands that must be routed in the lower layer. The problem allows the traffic for each demand to be split among different paths. The different layers represent different technologies that are used conjointly in telecommunications networks, for instance, SDH over WDM (Pióro and Medhi, 2004).

Second, we study the hop-constrained path-diversified network design problem, (**HOP**). Given a set of point-to-point demands, a graph, a hop limit index L, and a number of paths K, (**HOP**) consists of choosing links of the graph so that the resulting network is able to route each commodity through a set of K disjoint paths that do not contain more than L links (or hops). The model does not consider link capacities so that, in particular, every link included in the network can route an arbitrary number of different commodities. The minimum number of paths K models the traffic protection, while the maximum number of hops L limits the delay along each of the used path.

Electrical power industry

The transmission expansion planning problem arises from the growth of energy demand over the years, yielding necessary changes in the electrical system. Namely, new generators should be built in order to meet the increasing needs in electrical power. Ideally, we would like to build new generating units to tailor the supply of nearby consumers. However, it is usually not possible or not economical to build the new generating units close to consumption centers so that they must be constructed in distant places. Brazil, for instance, relies heavily on hydropower whose generating units are usually located far from main cities and industries. Therefore, it is necessary to build new transmission circuits in order to integrate all power plants into the electrical grid.

The decisions of the planning process can be divided in choosing the best generating units and their emplacements, and the best routes of transmission. This decision process leads to an optimization problem of great size that must be solved by planning engineers. They need to develop strategies and techniques to ensure that decisions made during the planning process are either optimal decisions or at least economically close to the optimal decision. In this thesis, we focus on the problem of designing of the network, denoted by (**TEP**) in what follows, given that the new power plants are already built. Ideally, one would integrate both decisions into the same model but this would yield an untractable optimization problem.

We provide next an outline of the thesis and presents our main contributions.

1.2 Outline of the thesis

This thesis is divided into two parts plus a chapter introducing mathematical tools described next. The Chapter starts with a brief review of the main notations and mathematical objects used throughout the thesis. It then turns to a general model for network design problems, (**ND**), discusses general solution methods and makes a first general literature review of the problem. Finally, it explains how to extend (**ND**) to handle electrical power flows.

In the first part, we present specialized models and algorithms for network design problems (**ML**), (**HOP**), and (**TEP**). Then, Part 2 turns to stochastic models. Because of the complexity of models (**ML**), (**HOP**) and (**TEP**), the second part also studies an important substructure of (**ND**), the knapsack problem. Chapter 5 introduces a stochastic knapsack model and studies the problem both in the theoretical and numerical sides. Chapters 6 and 7 build on algorithmic and theoretical findings from Chapter 5. Namely, the LP/NLP algorithm presented in Chapter 5 is improved to take the network structure into account in Chapter 6, while the reasoning used to prove Theorem 5.9 is applied in Chapter 7 to linearize probabilistic constraints. We describe more precisely the contents of each chapter in what follows. Notice that each chapter starts with a literature review of the specific problem studied therein.

Chapter 3 studies models (**ML**) and (**HOP**). Although both models are complex already, they are still approximations of realistic problems. For instance, (**ML**) always allows fractional routing while (**HOP**) neglects links capacities. Both models are formulated as large MILPs, that we tackle through Benders decompositions. The novelty of our approach lies in a systematic study of when to generate cuts within branch-and-cut algorithms. We present a thorough computational study of various cutting plane and branch-and-cut algorithms on a large set of instances including the real based instances from SNDlib (Orlowski et al., 2007).

Chapter 4 studies the transmission network expansion planning problem (**TEP**). We study relevant theoretical and practical aspects of (**TEP**), set as a bilinear programming problem with mixed $0-1$ variables. We show that the problem is $\mathcal{NP}$-hard and that, unlike models (**ND**), (**ML**) and (**HOP**) from Chapters 2 and 3, a transmission network may become more efficient after cutting-off some of its circuits. For this reason, we introduce a new model that, rather than just adding capacity to the existing network, also allows for the network to be re-designed when it is expanded. We then turn into different reformulations of the problem, that replace the bilinear constraints by using a “big-M” approach. We show that computing the minimal values for the “big-M” coefficients involves finding the shortest and longest paths between two buses. We assess our theoretical results by making a thorough computational study on real electrical networks. The comparison of various models and reformulations shows that our new model, allowing for re-design, can lead to sensible cost reductions.

Chapter 5 studies the knapsack problem with weights and capacity following independent random variables and prove that the problem is weakly $\mathcal{NP}$-hard in general. We provide pseudo-polynomial algorithms for three special cases of the problem: constant weights and capacity uniformly distributed, subset sum with Gaussian weights and arbitrary random capacity, and subset sum with constant weights and arbitrary random capacity. We then turn to a branch-and-cut algorithm based on the outer approximation of the objective function. We provide computational results for the stochastic knapsack problem (i) with Gaussian weights and constant capacity and (ii) with constant weights and capacity uniformly distributed, on randomly generated instances inspired by computational results for the knapsack problem.

Many convex linearly constrained programs and mixed integer programs have a large number of variables, so that the variables should be generated dynamically throughout the solution algorithm. This yields to the well known “branch-and-price algorithm” and “simplicial decomposition”. We present in Chapter 6 a novel “branch-and-cut-and-price algorithm” to extend this idea to certain classes of convex linearly constrained MINLPs. Our algorithm incorporates the variable generation into the “LP/NLP algorithm” introduced by Quesada and Grossman and described in Chapter 5. We detail our framework for the stochastic network design problem with simple recourse.

Chapter 7 studies individual probabilistic linear constraints arising, for instance, when modeling unsplittable multi-commodity flow with uncertain demand. We show that such constraints can be linearized when all random coefficients are independently distributed according to either $\mathcal{N}(\mu_i, \lambda\mu_i)$, for some $\lambda > 0$ and $\mu_i > 0$, or $\Gamma(k_i, \theta)$ for some $\theta > 0$ and $k_i > 0$. The constraint can also be linearized when the coefficients are independent and identically distributed and either positive or strictly stable random variables.

Finally, the last chapter concludes with a summary of the thesis and proposes directions of future work.

2

Mathematical tools

In this chapter, we introduce most mathematical objects and notations used in this thesis. In particular, we use the following conventions. Models and formulations are denoted by bold letters such as (**ND**) and **SP**, algorithms to solve them by typewriter letters such as `cp`, discrete sets and their cardinalities are denoted by upper case sans serif-letters and $|*|$, such as B and $|\mathsf{B}|$, respectively, while continuous sets (including polyhedra) are denoted by upper case calligraphic letters, such as $\mathcal{B}$.

The next section introduces graph theoretic objects, Section 2.2 gives a quick introduction to integer programming and Section 2.3 formulates problem (**ND**) and explains how to extend the latter to handle electrical power flows.

2.1 Graph theory

Most of the problems studied in this thesis aim at designing networks. Given a set of nodes V, and a set of pairs of nodes E, we denote by $\mathsf{G} = (\mathsf{V}, \mathsf{E})$ the graph (or network) defined by V and E. An edge $e = ij = ji \in \mathsf{E}$ for $i, j \in \mathsf{V}$ represents an undirected link between nodes i and j. In what follows, node sets are always denoted by V, with sometimes an upper index, such as V^l. Similarly, edge sets are always denoted by E and F (possibly with an upper index) and edges are denoted by e, f or ij. When the direction of the link between i and j is relevant, we use arcs instead of edges, denoted by (i, j) and (j, i). Arc sets are denoted by A, possibly with an upper index. Given an undirected graph (V, E) we can trivially construct a bi-directed graph (V, A) by associating two arcs with each edge in E, one in each direction.

Graphs allow to model flows conveniently. For each arc $(i, j) \in \mathsf{A}$, $x_{ij} \geq 0$ denotes the amount of flow through arc (i, j). We define also the flow x_e on an edge e with head $t(e)$ and tail $s(e)$ (these are chosen arbitrarily); $x_e > 0$ indicates a positive flow from $s(e)$ to $t(e)$ while $x_e < 0$ indicates a positive flow from $t(e)$ to $s(e)$. When a capacity C_e is associated with an edge $e = ij$, the flow on e must not exceed C_e, that is, $-C_e \leq x_e \leq C_e$ or $x_{ij} + x_{ji} \leq C_e$. In network design problems, one must usually choose on which edges to install capacities. Installing capacity on an edge e has a cost $c_e \geq 0$, and binary variable y_e indicates whether the capacity is installed.

Flows in networks allow to model the shipment process of a set of commodities (data, energy, goods, ...) that must be displaced from a subset of nodes to another subset of nodes. This is modeled with the help of a demand vector $d \in \mathbb{R}^{|\mathsf{V}|}$. For each node i, demand d_i states whether node i must have an outgoing flow balance ($d_i > 0$), an ingoing flow balance ($d_i < 0$) or must only be used as a transhipment node ($d_i = 0$). When different commodities $q \in \mathsf{Q}$ transit by the same network, we associate a vector $d^q \in \mathbb{R}^{|\mathsf{V}|}$ for each commodity $q \in \mathsf{Q}$.

2.2 Integer programming

Mathematics can help in solving many complex planning and optimization problems. In particular, mathematical programming models the optimization problem with the help of functions $h, g_i : \mathbb{R}^{m+n} \to \mathbb{R}$, $i = 1, \ldots, l$, yielding what is often called a mathematical program

$$\min\{h(x,y) \text{ s.t. } g_i(x,y) \leq 0, x \in \mathbb{R}^m, y \in \mathbb{Z}^n\}, \tag{2.1}$$

or a mixed-integer program (MIP) when $m > 0$ and $n > 0$. This thesis takes a closer look at linear MIPs, that consider only linear functions h and g_i, and convex and differentiable MIPs, that consider only convex and differentiable functions h and g_i. Trivially, any linear MIP is also a convex and differentiable MIP.

2.2.1 Complexity of optimization problems

In what follows, we provide a brief introduction to the complexity theory of optimization problems, see Wolsey (1998); Garey and Johnson (1979) for further details. To define the complexity of optimization problem (**P**), we must first associate the following decision problem (**D**) to (**P**). Given a threshold k, (**D**) asks whether the cost of the optimal solution to (**P**) is below k. Problem (**D**) is in $\mathcal{NP}$ if we can verify in polynomial time whether a candidate vector provides the answer to (**D**) (see Garey and Johnson (1979) for a rigorous definition of polynomial time). If, moreover, every problem in $\mathcal{NP}$ can be reduced to (**D**) in a polynomial number of operations, (**D**) is $\mathcal{NP}$-complete and the associated optimization problem (**P**) is $\mathcal{NP}$-hard. For example, (2.1) is a $\mathcal{NP}$-hard problem when g_i and h are linear. However, for general g_i and h, the decision problem associated to (2.1) is undecidable (Jeroslow, 1973), making (2.1) more complex than $\mathcal{NP}$-hard problems. Easy problems in $\mathcal{NP}$ are those belonging to subset $\mathcal{P} \subseteq \mathcal{NP}$. An optimization problem (**P**) is in $\mathcal{P}$ (or *polynomially solvable*) if we can find an algorithm that can solve any instance of the problem in polynomial time.

In theory, it is not known whether $\mathcal{P}$ and $\mathcal{NP}$ are different sets. Nevertheless, $\mathcal{P} \neq \mathcal{NP}$ is a working hypothesis. Namely, once a problem is proved to be $\mathcal{NP}$-hard, we do not expect that there exists an algorithm to solve the problem in polynomial time. For this reason, polynomially solvable optimization problems are usually said to be easy, while $\mathcal{NP}$-hard problems are said to be difficult. Note, however, that this is only an indication on the difficulty of the problem because only the worst-case complexity of the algorithm is considered, not its expected behavior. Then, although most real-life optimization problems are $\mathcal{NP}$-hard, some of them are significantly harder to solve than others, which is not reflected by their complexity. For instance, although network design problems studied in Chapters 3 and 4 are all strongly $\mathcal{NP}$-hard, we are able to solve those from Chapter 4 for graphs two or three times larger than those used in Chapter 3.

2.2.2 Solution methods

For linear MIPs, we can replace (2.1) by the following description

$$(\mathbf{MILP}) \qquad \begin{array}{rl} \min & c^t x + k^t y \\ \text{s.t.} & Ax + By \geq e \\ & x \geq 0,\ y \geq 0 \text{ and integer,} \end{array}$$

where x, y are the variables of the model, and $c \in \mathbb{R}^m, k \in \mathbb{R}^n, e \in \mathbb{R}^l, A \in \mathbb{R}^{lm}, B \in \mathbb{R}^{ln}$ are the parameters. Problem (**MILP**) is, in general, $\mathcal{NP}$-hard, so that enumeration-type algorithms should be considered (unless $\mathcal{P} = \mathcal{NP}$). When $n = 0$, (**MILP**) does not contain integer restrictions and the problem is called a linear program. Although linear programs are polynomially solvable, they are most often solved by the simplex algorithm, that has an exponential worst-case running time. This is another example of the fact that “polynomial” algorithms are not always faster in practice than “exponential” algorithms. When $n > 0$ but the integrality requirement on y is relaxed, we obtain (**LP**), the linear programming relaxation of (**MILP**).

Modern developments in integer programming allow (**MILP**) to be solved efficiently by standard solvers for a large class of problems. These solvers implement highly tuned versions of branch-and-cut algorithms. Branch-and-cut algorithms are clever enumeration algorithms that rely on (**LP**) and heuristics to bound the solution of (**MILP**)from below and above, respectively. First, they solve (**LP**). If the solution is integer, it is the solution to (**MILP**). Otherwise, the integrality is recovered by branching or adding cutting planes. Given a fractional solution to (**LP**), the branching procedure creates two or more subproblems by partitioning the original domain of (**MILP**), while a cutting plane is a linear inequality violated by the current fractional solution and satisfied by all vectors feasible for (**MILP**). Besides branching and cutting, branch-and-cut algorithms also rely on fathoming some of the subproblems using upper and lower bounds, avoiding solving all of them. More details on methods to solve (**MILP**) are presented in Section 2.3.2, specialized to network design problems.

When g_i are linear functions and h is a non-linear convex and differentiable function, we show below how to formulate the problem as (**MILP**), given that h and all variables are bounded. This procedure, called *outer approximation*, begins with the rewriting (2.1) as the following infinite mixed-integer linear program:

$$
(\textbf{MINLP}) \quad
\begin{array}{rl}
\min & \gamma \\
\text{s.t.} & Ax + By \geq e \\
& \gamma \geq h(\overline{x},\overline{y}) + \sum_{i=1}^{m} \frac{\partial h}{\partial x_i}(\overline{x},\overline{y})(x_i - \overline{x}_i) + \sum_{i=1}^{n} \frac{\partial h}{\partial y_i}(\overline{x},\overline{y})(y_i - \overline{y}_i) \\
& \qquad \overline{x} \in \mathbb{R}^n, \overline{y} \in \mathbb{R}^m \quad (2.2) \\
& \gamma \geq 0,\ x \geq 0,\ y \geq 0 \text{ and integer,}
\end{array}
$$

where h has been substituted by (2.2) that describes tangent hyperplanes for all $\overline{x} \in \mathbb{R}^n, \overline{y} \in \mathbb{R}^m$. Let $\epsilon > 0$ be small. Since h and all variables are bounded, we can select a finite subset of (2.2) that yield a piece-wise approximation of h whose maximum distance from h is less than ϵ. Therefore, the optimal solution of the resulting problem (with finitely many equations) is at most ϵ less than the optimal solution of (**MINLP**). This process of replacing a non-linear and convex function by a piece-wise tangent approximation is called outer approximation. Non-linear convex and differentiable g_i can be approximated similarly. Therefore, convex and differentiable MINLPs can also be solved by branch-and-cut algorithms, as those developed in this thesis. Still, how and when to generate these tangent hyperplanes is not a trivial question if we do not want to end up with very large formulations. Notice that than different solution methods exist to handle (**MINLP**), that do not rely on an outer-approximation of the feasible set, see the review from Grossmann (2002). These techniques are especially important when h or some of the g_i are non-convex.

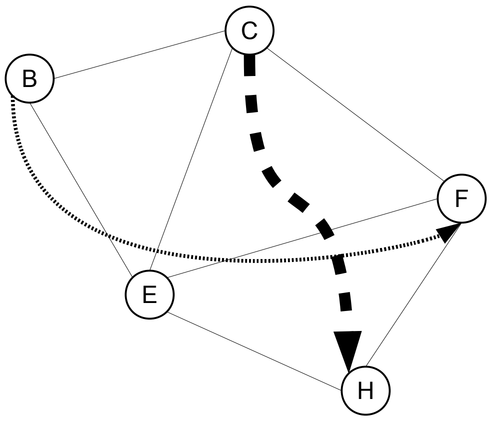
Next section takes a closer look at network design models.

2.3 Network design models

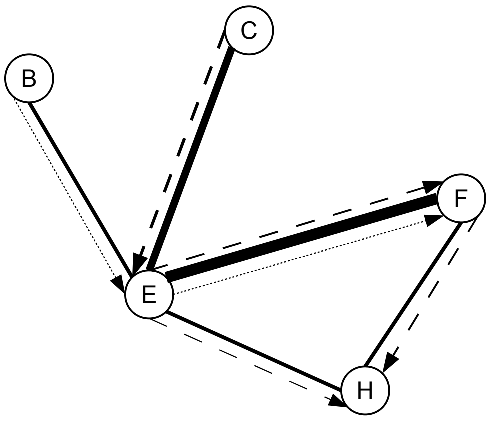
In this section, we formulate a general network design problem and review methods to solve the the problem. We show then how to extend the model in order to handle electrical power flows.

2.3.1 Formulation

Given an undirected graph (V, E) and a set of commodities Q, with origin $s(q)$, destination $t(q)$, and nominal value d^q for every $q \in \mathsf{Q}$, the capacitated network design problem aims at installing the cheapest capacities on edges of G so that the resulting network shall be able to attend to each demand. Each edge $e \in \mathsf{E}$ between i and j can be used in both directions, so that we can introduce the set of arcs A



(a) We are given a graph and two commodities.



(b) We must fix the capacity on each edge and find a routing for each commodity.

Figure 2.1: Network design example with $|\mathsf{Q}| = 2$.

making (V, A) a bi-directed graph. Integer variable y_e states how many batches of capacity C and cost c_e are installed on edge e, while fractional variable x^q_{ij} describes the amount of flow for commodity q through arc (i, j). The model reads as follows:

$$\min \quad \sum_{e \in \mathsf{E}} c_e y_e \tag{2.3}$$

$$\text{s.t.} \quad \sum_{q \in \mathsf{Q}} \left(x^q_{ij} + x^q_{ji} \right) \le C y_e \qquad e = ij \in \mathsf{E} \tag{2.4}$$

$$(\mathbf{ND}) \qquad \sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}} x^q_{ji} - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}} x^q_{ij} = \begin{cases} -d^q & \text{if } i = s(q) \\ d^q & \text{if } i = t(q) \\ 0 & \text{else} \end{cases} \qquad i \in \mathsf{V},\, q \in \mathsf{Q} \tag{2.5}$$

$$x \ge 0$$

$$y \ge 0 \text{ and integer.} \tag{2.6}$$

Objective (2.3) minimizes design cost, constraints (2.4) ensure that capacities are not exceeded (note that flows in both directions of an edge e share the same capacity Cy_e), constraints (2.5) state that, for each commodity, the outgoing flow at every node of the graph must be equal to the ingoing flow except for extremities $s(q)$ and $t(q)$. Finally, (2.6) ensures that capacities are installed by batches. Problem (**ND**) is illustrated in Figure 2.1. We are given the graph from Figure 2.1(a) with two commodities. Then, we decide to install the capacities y represented by the plain links in Figure 2.1(b) and to use the routings x represented by the dotted and dashed lines.

Problem (**ND**) is most often called Network Design Problem, or Network Loading Problem, and has been studied extensively for many years (Yuan, 2001). It is usually used to model telecommunications networks, because each commodity q has a unique source $s(q)$ and a unique destination $t(q)$. To extend (**ND**) to more general commodities, such as goods in service network design (Crainic, 2000), or electrical power in transmission network expansion (see Section 2.3.3), we may replace (2.5) by

$$\sum_{j \in \mathsf{V}:(j,i)\in \mathsf{A}} x_{ji}^q - \sum_{j \in \mathsf{V}:(i,j)\in \mathsf{A}} x_{ij}^q = d_i^q \qquad i \in \mathsf{V},\, q \in \mathsf{Q}, \tag{2.7}$$

where d_i^q is either the supply ($d_i^q < 0$) or the demand ($d_i^q > 0$) at node i. It is easy to see that, each vector d^q must satisfy $\sum_{i\in V} d_i^q = 0$ for (2.7) to describe a feasible problem. Quite often, additional side constraints are required to model accurately the problem. Some examples will be given in the first part of this thesis.

Other variations of (**ND**) add a routing cost to (2.3), considering for instance travel time, relax integrality constraints (2.6), or force x to be integer. In fact, many applications in telecommunications require that each commodity be routed along a unique path. This is modeled by replacing d^q by 1 in (2.5), by forcing x to be binary and by replacing (2.4) by the knapsack constraint

$$\sum_{q\in \mathsf{Q}} d^q \left(x_{ij}^q + x_{ji}^q\right) \le C y_e \qquad e = ij \in \mathsf{E}. \tag{2.8}$$

Equation (2.8) with binary x hardens (**ND**) substantially by adding a large number of binary variables to the problem. Nevertheless, the structure provided by (2.8) may also be exploited advantageously, see Chapters 5 and 7.

Another important aspect of (**ND**) is the prediction of the exact value of the nominal demands. In many situations, it is not possible to predict d^q (or d_i^q) exactly at the time design decisions must be made. This gives rise to robust and stochastic models, where exact values of d^q and d_i^q are replaced by polyhedra or random variables. In the second part of this thesis, we shall take a closer look at the case of random variables.

2.3.2 Solution methods

Capacitated network design problems such as (**ND**) are still hard to solve for many real-sized networks. The two main difficulties of (**ND**) arise from:

1. capacity constraints (2.4) yielding a weak LP relaxation;

2. the number of flow conservation constraints (2.5), growing rapidly with the problem size.

Lower bound improvement

In order to reinforce the (weak) LP relaxation of (**ND**), many researchers have looked at strong cutting planes, including Atamturk (2002); Bienstock and Gunluk (1996); Bienstock et al. (1998); Dahl and Stoer (1998); Raack et al. (2011). More recently, Achterberg and Raack (2010) have developed a procedure to automatically detect a network design structure within a general MILP, generating strong cutting planes accordingly. Although useful valid inequalities are presented in Chapter 4, the methods studied in this thesis are more closely related to decomposition methods handling the size of (**ND**), as explained below.

Another line of research in reducing the large gap of (**ND**) considers extended formulations. The latter rewrite the problem with a very large number of binary variables in order to obtain a tight formulation (Frangioni and Gendron, 2009; Ljubic et al., 2009).

Decomposition

Most common decomposition methods belong to one of the following: Benders decomposition, Dantzig-Wolfe decomposition and Lagrangian relaxation. A common feature of the three methods when applied to (**ND**) is that they break the structure of the problem into many subproblems (one per commodity or one per edge). Doing so usually reduces solution time because most algorithms solve MILP and LP in an amount of time increasing faster than linearly with the size of these problems. Put simply, it is in general much faster to solve 10 problems of a given size n than to solve a single problem of size $10n$. In fact, practical situations are far from being this simple. Solution algorithms implemented within commercial MILP and LP solvers are increasingly able to detect the particular structure of a problem and to use it for pre-processing, cutting planes generation and heuristics, among others. Once we decompose the problem, the solver has a weaker understanding of the structure and becomes less efficient. Therefore, when we decompose a problem, we somehow assume that we shall be able to use the structure of the problem more efficiently than the solver would. Fortunately this is true is most cases, which is why decomposition methods often give very good results.

Sometimes decomposition methods must be used because we do not have enough memory to solve the whole problem at once. This is especially true in Stochastic Programming, where large scenario sets make the problem large-scale. Another reason for using decomposition methods (different from Benders decomposition) is that they may yield a tighter bound than the LP relaxation. Specifically, we can choose to transfer some of the complexity to the subproblems.

Benders decomposition is one of the most common decomposition for (**ND**) (Gabrel et al., 1999; Costa, 2005), maybe because it is very easy to implement. However, it suffers from two drawbacks. First, as previously mentionned, it can not improve the bound within enumeration algorithms. Second, it is in general impossible to project out integer variables, and thus we can not apply that methodology to problems with integer routing. Dantzig-Wolfe Decomposition has been used for (**ND**) by Frangioni and Gendron (2010), among others, and for the closely related multi-commodity flow problem by Barnhart et al. (2000), among others. Finally, Lagrangian decomposition has been used by Crainic et al. (2001), among others, together with a bundle algorithm.

This thesis relies heavily on decomposition methods, more specifically, Benders and Dantzig-Wolfe decompositions, see Chapters 3 and 6, respectively.

2.3.3 Power transmission networks

Model (**ND**) needs two significant refinements to describe the design of electrical power transmission networks. As discussed already, flow balance constraint (2.5) must be replaced by the more general (2.7) for a unique commodity. Then, the flow in an electrical power network must satisfy to physical laws. The widely used *DC* model supposes that the flow x_{ij} between two nodes (or buses) i and j must satisfy to

$$x_{ij} = \psi_{ij}(\theta_i - \theta_j), \tag{2.9}$$

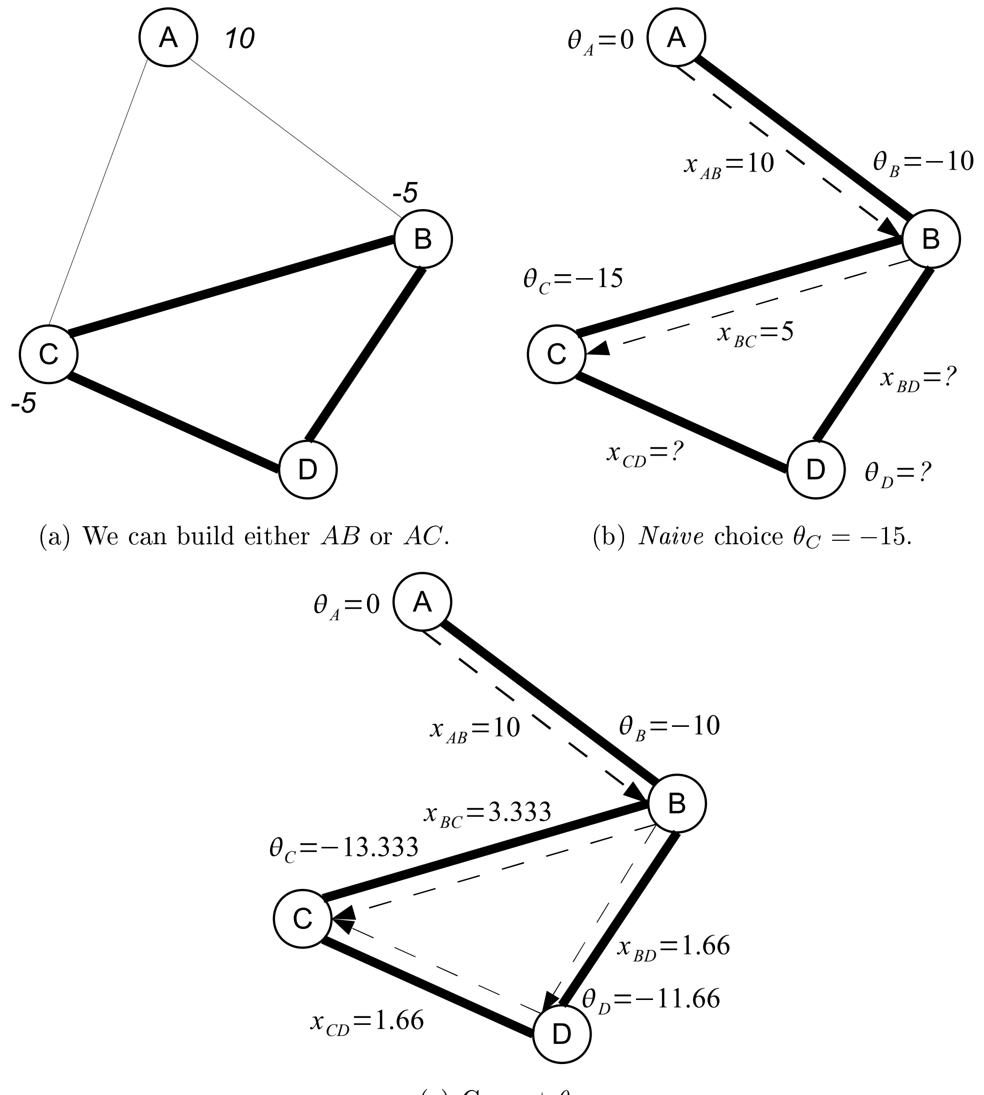
where ψ_{ij} is the reactance of edge (or circuit) $ij \in \mathsf{E}$ and θ_i is the potential angle at node i. Note that ψ is a vector of parameters while θ contains decisions variables of the problem. Equation (2.9) implies that flow variables are not needed to formulate electrical power flows problems because they are induced by potential differences, which are variables of the problems. Nevertheless, they are included into the formulations of such problems to ease understanding. For instance, constraint $x_{ij} \leq C_{ij}$ that limits the amount of power flow on a circuit ij is clearer than the equivalent $\psi_{ij}(\theta_i - \theta_j) \leq C_{ij}$.

Let us explain equation (2.9) by the example shown in Figure 2.2. Physical units are not written for the sake of simplicity. We are given the network depicted in Figure 2.2(a), with three existing circuits. To attend the demand of nodes B and C, generator A should be connected to the transmission network. Assume that we choose to build the circuit between A and B. Potential angle θ_B must be set to -10 to convey all the electrical power from A to B. The values of θ_C and θ_D are less straightforward. Imagine that we set *naively* θ_C to -15 to convey 5 units of flow from B to C, see Figure 2.2(b). What shall be the flows on BD and CD ? Although we would like these flows to be null, they will depend on the value of θ_D. In fact, the correct values for θ_C and θ_D are the solutions to the linear system of equations below, see also Figure 2.2(c):

$$\begin{array}{rrrclllr}
-x_{BC} & -x_{BD} & & = & & & & -5 \\
x_{BC} & & -x_{CD} & = & & & & 5 \\
& x_{BD} & +x_{CD} & = & & & & 0 \\
x_{BC} & & & = & \theta_B & -\theta_C & & \\
& x_{BD} & & = & \theta_B & & -\theta_D & \\
& & x_{CD} & = & & \theta_C & -\theta_D. &
\end{array}$$

Introducing design variable y_{ij} stating whether circuit $ij \in \mathsf{E}$ is built or not, we have that x_{ij} must be equal to 0 when y_{ij} is equal to 0, and equal to $\psi_{ij}(\theta_i - \theta_j)$ when y_{ij} is equal to 1. This disjunction is modeled through the bilinear constraint

$$x_{ij} = \psi_{ij} y_{ij}(\theta_i - \theta_j). \tag{2.10}$$



(a) We can build either AB or AC.

(b) *Naive* choice $\theta_C = -15$.

(c) Correct θ.

Figure 2.2: Transmission network expansion example with $\psi_{ij} = 1$ and $C_{ij} > 10$ for each $ij \in \mathsf{E}$.

More details about the practical implications of (2.9) and (2.10) are given in Chapter 4.

The refinements mentioned above provide telecommunications and power transmission network design problems with different structures. For instance, the size of (**ND**) increases rapidly with the number of commodities $|\mathsf{Q}|$, making decomposition methods necessary to solve real-life problems. In opposition, transmission networks route a sole commodity so that the size of the formulation is usually not the main difficulty. In fact, the main difficulty of transmission network expansion problems arises from (2.10), which must be linearized and yield a very weak linear programming relaxation. The resulting formulation contains little combinatorial structure, so that it is a hard task to derive efficient valid cutting planes and extended formulations. We provide an extensive literature review of methods to solve the transmission expansion planning problem in Chapter 4.

Part I

Network design problems

3

Benders decomposition for telecommunications network design

3.1 Introduction

Our society is increasingly dependent on large-scale, networked information systems of remarkable scope and complexity. Telecommunication networks are designed with a layered structure, according to different technologies. For instance, one could consider a virtual layer over a physical layer, also called transport layer. This leads to bi-layer network design problems. In those problems, demands are usually given in the virtual layer. They have to be routed in the virtual layer, leading to the installation of "virtual capacities" (which are routers or other devices). Virtual capacities define demands for the transport layer, leading to the installation of capacities (optical fiber, copper link, ...), in the physical layer. Therefore, when a demand is routed through a path in the virtual layer (composed of many virtual edges), each edge corresponds to a path in the layer underneath (also called a "grooming path").

Technically, each layer has its own technology (Pióro and Medhi, 2004), for instance:

- MPLS: Multi-Protocol Label Switching,
- WDM: Wavelength-Division-Multiplexing,
- SDH: Synchronous Digital Hierarchy.

Incorporating survivability capabilities into a network has become unavoidable for network operators in order to mitigate the risks in case of failures. Herein, survivability is the capability of a system to fulfil its mission in a timely manner despite intrusions, failures, or accidents. This concept of survivability allows networks to remain functional when links or nodes fail. For each demand, we impose that at least K different paths exist for each origin-destination pair. These K paths can, for instance, be "edge-disjoint", i.e. if a particular edge belongs to one of the paths, this particular edge can not be used by the other paths. This guarantees that if $K-1$ edges break down, it is always possible to reroute all the demands by the K-th path which does not use the broken arcs.

In general, the survivability constraints are not sufficient to guarantee a cost effective routing with a good quality of service since the primary path where the traffic flows or the secondary paths, if a path fails, may lead to unacceptable delays. Since in most of the routing technologies, delay is caused at the switching nodes because node processing times dominate over queuing delays, it is usual to measure the delay in a path in terms of its number of intermediate nodes, or equivalently, its number of arcs (or hops). Thus, to guarantee the required quality of service, we impose a limit on the number of arcs of the routing paths, so that the traffic may be routed, or rerouted if a path fails, on a different path with a quality of service guaranteed in terms of delay. These so-called hop constraints also guarantee a certain level of transmission reliability in the sense that the probability that all the transmission lines in the path are working decrease with the number of arcs (Woolston and Albin, 1988). We say that a L-path is a path with at most L arcs (or hops).

3.1.1 Contributions and structure of the chapter

Grouping all considerations into a single model would be untractable in a computational point of view. Thus, we consider the following two models:

- A bilayer network design problem, where capacities must be installed in each layer. Neither survivability nor hop-constraints are considered.

- A single layer hop-constraint path-diversified network design problem, where we neglect edges' capacities. Namely, a fixed cost is paid to use an edge, regardless to the capacity required.

However, both models being particular cases of *Fixed-Charge Network Design Problems*, they have characteristics in common. For instance, both must route multi-commodity flows and thus, a modeled by large-scale linear formulations for which decomposition algorithms such as the Benders decomposition are well suited. This motivates the grouping of both problems into a single chapter. Our main achievement is a careful implementation of the Benders decomposition within a branch-and-cut algorithm, obtaining significantly better results than previous works on the problems. Some recent works of Fischetti et al. (2010) and Ljubic et al. (2009), among others, have highlighted the importance of the normalization constraint in the separation problem. Herein, we investigate another aspect of the algorithm, namely, when to generate cuts. We present a thorough computational study of various cutting plane and branch-and-cut algorithms on a large set of instances including the real based instances from SNDlib (Orlowski et al., 2007).

Multi-layer network design

The work presented in the following has been published in Fortz and Poss (2009). We improve the constraint generation method used by Knippel and Lardeux (2007). Our branch-and-cut algorithm solves the Benders decomposition of the problem more than 10 times faster on average than the cutting planes from Knippel and Lardeux (2007).

Hop-constrained path diversified network design

The work presented in the following is described in details in Botton (2010) and Fortz et al. (2010). Herein, we focus on the computational results only. We formulate the network design problem for any $L, K \geq 1$ with multiple pairs of terminals $(s(q), t(q))$, for $q \in \mathsf{Q}$, as an integer program based on the layered representation from Gouveia (1998). Up to our knowledge, this is the first formulation for the problem valid for $L \geq 5$ and any $K \geq 1$. For $L = 2, 3$ our branch-and-cut algorithms separate at the same time (and polynomially) "cut inequalities" and "L-path inequalities" while the branch-and-cut algorithm from Huygens et al. (2007) needs to separate both independently. Finally, we present a fast and efficient LP-based heuristic that provides the optimal solution for more than half of the instances tested.

In Section 3.1.2, we review most important previous works on the problems, see Fortz et al. (2010) and Botton (2010) for a more thorough literature review. Sections 3.2 describes extended formulations for both problems. The following section recall the basics of Benders decomposition, before reformulating both problems from Section 3.2. The main contribution of our work, the algorithms, is presented in Section 3.4. We describe various cutting planes and branch-and-cut algorithms as well as a heuristic. Finally, Section 3.5 provides an extensive computational study of the algorithms.

3.1.2 Literature review

We present next a litterature review for the two network design problems studied in this chapter.

Multi-layer network design

As the single-layer capacitated network design problem is complicated enough, most approaches for the bi-layer problem consider each layer separately:

- First, a network design problem is solved for the virtual layer only.
- Then, virtual capacities to be installed in the virtual layer define demands for another network design problem, for the transport layer this time.

Though much easier to solve, this relaxed approach might provide solutions far from the optimal solution of the problem. Therefore, an integrated approach should be considered.

Network design has been widely studied for many years (Yuan, 2001). However, the interest in multi-layered network design is more recent and can be traced back to a paper by Dahl et al. (1999). They assume given physical capacities and aim to select virtual edges (called pipes in the paper) and to configure the routing in both layers. A polyhedral study is made resulting in a cutting-plane algorithm. Since then, the interest in this field has rapidly grown and different approaches have been suggested to address these problems.

Orlowski and Wessäly (2004) begin by giving a good introduction to multi-layered networks where they offer technical examples and develop a model considering many technical constraints. However, they do not propose a specific solution method. In a further paper with Koster et al., different branch-and-cut approaches are developped. Extending previous work by Belotti and Malucelli (2005), the authors briefly describe a cut-and-branch-and-price algorithm. Then, they solve an integer formulation using a branch-and-cut framework (Koster et al., 2007), where they introduce efficient heuristics. Finally, they address a more complex formulation, taking node hardware and survivability into account (Baier et al., 2007). They also extend and test different sorts of cuts coming from mono-layer models (Raack et al., 2011).

Belotti et al. (2006) and Capone et al. (2007) study multi-layered design with statistical multiplexing, the motivation being that routing different commodities on the same capacity results in less variation of the flow on the capacity. They compute a lower bound through a Lagrangian relaxation and use heuristics to find good upper bounds.

Kubilinskas and Pióro (2005) address the problem of maximizing the profit of satisfying demands in a bi-layer (MPLS over WDM) situation. They present an iterative procedure to solve their complex mixed-integer problem. This procedure consists of splitting the problems into two stages, one for each layer, where the solution of the first layer defines demands in the second one. Then the routing solution in the physical layer leads to cost modification for edges in the first layer and the whole problem is solved again.

Holler and Voss (2006) propose an integer programming formulation for two layer networks consisting of SDH over WDM. Strictly speaking, this is not a multi-layer problem in the sense that demands are routed through either SDH links or WDM links. They solve the problem using two different heuristics.

Gabrel et al. (1999) introduce a constraint generation procedure based on a Benders decomposition for capacitated network design problems. Knippel and Lardeux (2007) and Geffard et al. (2007) extend this work to multi-layered networks and multi-period time scheduling, respectively. In Knippel et al. (2003), the authors improve these methods using the knapsack-like structure of the master problem to facilitate its resolution.

Hop-constrained path diversified network design

The L-hop constrained network design problem, consists in finding a least cost subgraph of G such that there exists a L-path between every pair of nodes. This problem was studied by Balakrishnan and Altinkemer (1992) as a means of generating alternative base solutions for a network design problem. They presented a standard network flow formulation with an additional cardinality constraint for each commodity (to model the hop constraints). They have also derived a Lagrangian relaxation based method whose theoretical best bound improves the linear programming bound given by the previous formulation. Later on, in the context of a directed spanning tree problem (which can be seen as a special case of a single-source L-hop-constrained network design problem), Gouveia (1998) presented a layered network flow reformulation whose linear programming bound equals a lagrangean based bound similar to the one proposed by Balakrishnan and Altinkemer. From then on, the reformulation has been used in several network design problems with hop constraints (e.g, Pirkul and Soni (2003), Gouveia and Magnanti (2003) and Gouveia et al. (2003)) and even some hop-constrained problems involving survivability considerations.

Formulating the problem in the space of the natural variables, that is, without using the layered formulation is not an easy task For $K = 1$, Dahl (1999) has provided such a formulation and shown that it describes the corresponding convex hull for $L \leq 3$. Later on, Dahl et al. (2004) have shown that finding such a description for $L \geq 4$ would be quite more complicated. For $K \geq 2$ the results are even worse. Huygens et al. (2004) have extended Dahl's result for $K = 2$ and $L \leq 3$. For $L \geq 4$, the only interesting result for the moment is the one from Huygens and Mahjoub (2007) for $L = 4$ and $K = 2$ where a valid formulation has been given. However, nothing else is known which explains that the only cutting plane method for the more general problem with several sources and several destinations, Huygens et al. (2007) only considers $L \leq 3$.

3.2 Models

We describe now an integer programming formulation for each problem.

3.2.1 Multi-layer network design

The model described next minimizes the cost of capacities installed in both layers. There is no cost associated with the routing. First, we must route commodities given by the demand matrix in the upper layer. This results in the installation of capacities in that layer. Then, each upper edge defines a commodity in the lower layer with a demand equal to the capacity installed on the upper edge.

Hence, there is a strong interaction between the two layers. Two feasible solutions with the same upper layer cost can have different overall costs since the cost of lower capacities can differ. This model is therefore more complex than the single layer capacitated network design model.

The two layers are represented by undirected graphs $\mathsf{G}^u = (\mathsf{V}, \mathsf{E}^u)$ and $\mathsf{G}^l = (\mathsf{V}, \mathsf{E}^l)$ for the upper and lower layer, respectively, constructed on the same node set V. To define the flows, it is convenient to introduce two directed arcs for each edge, yielding sets of arcs A^l and A^u. For $i \neq j \in \mathsf{V}$, we denote the directed arc from i to j in A^l or A^u by (i,j), the undirected edge in E^l by ij or e, and the undirected edge in E^u by ij or f. Each commodity $q \in \mathsf{Q}$ to be routed in the upper layer from $s(q)$ to $t(q)$ must route an amount equal to d^q.

Our model uses a node-arc formulation for each layer. The objective (3.1) minimizes the sum of the costs c^l_e (resp. c^u_f) of the y^l_e (resp. y^u_f) modules of capacity that are installed in the lower layer (resp. upper layer), with modular capacity C^l (resp. C^u). Then, variable x^{uq}_{ij} specifies the amount flowing through arc $(i,j) \in \mathsf{A}^u$ for commodity $q \in \mathsf{Q}$.

Recall that commodities to be routed on the lower layer are given by capacities installed in the upper layer. Therefore, the set of commodities in the lower layer is identified with the set of (undirected) upper edges E^u so that for each edge $f \in \mathsf{E}^u$, we choose arbitrarily one of the extremities as the source $s(f)$ and the other as the destination $t(f)$. Variable x^{lf}_{ij} specifies the flow on arc $(i,j) \in \mathsf{A}^l$, related to upper capacity Cy^u_f.

With this set of variables, the problem can be formulated as:

(ML)

$$\min \sum_{e \in \mathsf{E}^l} c^l_e y^l_e + \sum_{f \in \mathsf{E}^u} c^u_f y^u_f \tag{3.1}$$

$$\text{s.t.} \sum_{q \in \mathsf{Q}} \left(x^{uq}_{ij} + x^{uq}_{ji}\right) \le C^u y^u_f \qquad f = ij \in \mathsf{E}^u \tag{3.2}$$

$$\sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}^u} x^{uq}_{ji} - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}^u} x^{uq}_{ij} = \begin{cases} -d^q & \text{if } i = s(q) \\ d^q & \text{if } i = t(q) \\ 0 & \text{otherwise} \end{cases} \qquad i \in \mathsf{V},\ q \in \mathsf{Q} \tag{3.3}$$

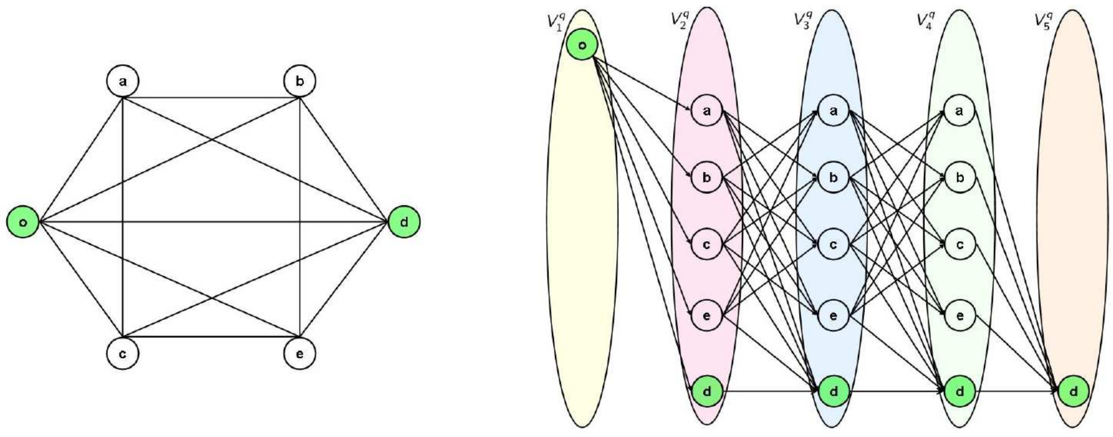
$$\sum_{f \in \mathsf{E}^u} \left(x^{lf}_{ij} + x^{lf}_{ji}\right) \le C^l y^l_e \qquad e = ij \in \mathsf{E}^l \tag{3.4}$$

$$\sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}^l} x^{lf}_{ji} - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}^l} x^{lf}_{ij} = \begin{cases} -C^u y^u_f & \text{if } i = s(f) \\ C^u y^u_f & \text{if } i = t(f) \\ 0 & \text{otherwise} \end{cases} \qquad i \in \mathsf{V},\ f \in \mathsf{E}^u \tag{3.5}$$

$$x^u,\ x^l \ge 0 \tag{3.6}$$

$$y^u,\ y^l \ge 0 \text{ and integer,} \tag{3.7}$$

yielding a model with a structure similar to (**ND**) for each layer. Constraints (3.2) and (3.4) impose that the total flow on an edge is less than the capacity installed



(a) (b)

Figure 3.1: Basic Network (a) and its Layered Representation (b) when $L = 4$

on that edge, whereas (3.3) and (3.5) ensure that the flow balance at each node is satisfied. Integrality constraints (3.7) force capacities to be installed by batches. Finally, because routing variables are continuous (3.6), each commodity can be split among an arbitrary number of paths in each layer.

3.2.2 Hop-constrained path diversified network design

The main idea of Gouveia (1998) is to model the subproblem associated with each commodity with a directed graph composed of $L + 1$ layers as illustrated in Figure 3.1. Namely, from the original undirected graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$, we create a directed layered graph $\mathsf{G}^q = (\mathsf{V}^q, \mathsf{A}^q)$ for each commodity, where $\mathsf{V}^q = \mathsf{V}_1^q \cup \ldots \cup \mathsf{V}_{L+1}^q$ with $\mathsf{V}_1^q = \{s(q)\}$, $\mathsf{V}_{L+1}^q = \{t(q)\}$ and $\mathsf{V}_l^q = \mathsf{V} \backslash \{s(q)\}$, $l = 2, \ldots, L$. Let v_l^q be the copy of $v \in \mathsf{V}$ in the $l - th$ layer of graph G^q, $s(q) = s(q)_1^q$ and $t(q) = t(q)_{L+1}^q$. Then, the arcs sets are defined by $\mathsf{A}^q = \{(i_l^q, j_{l+1}^q) \mid ij \in \mathsf{E}, i_l^q \in \mathsf{V}_l^q, j_{l+1}^q \in \mathsf{V}_{l+1}^q, l \in \{1, \ldots, L\}\} \cup \{(d(q)^l, d(q)^{l+1}) \mid l \in \{1, \ldots, L\}\}$, see Figure 3.1. In what follows, an (undirected) edge in E is denoted ij or e while a (directed) arc between $i_l^q \in \mathsf{V}_l^q$ and $j_{l+1}^q \in \mathsf{V}_{l+1}^q$ is denoted by (i, j, l) (the commodity q is omitted in the notation as it is often clear from the context).

Note that each path between $s(q)$ and $t(q)$ in layered graph G^q is composed of exactly L arcs (hops), which corresponds to a maximum of L edges (hops) in the original one. In fact this is the main idea of this transformation since in the layered graph, any path is feasible with respect to the hop constraints. The usual network flow equations defined in this layered graph yield the following model:

$$
\text{(HOP)}\quad
\begin{aligned}
\min\quad & \sum_{e\in \mathsf{E}} c_e y_e \\
\text{s.t.}\quad & \sum_{l\in\{1,\dots,L\}} \left(x_{ij}^{lq} + x_{ji}^{lq}\right) \le y_e \qquad e\in\mathsf{E},\ q\in\mathsf{Q} && (3.8)\\
& \sum_{j:(j,i,l-1)\in\mathsf{A}^q} x_{ji}^{l-1q} - \sum_{j:(i,j,l)\in\mathsf{A}^q} x_{ij}^{lq} = \begin{cases} -K & \text{if } i = s(q)\\ K & \text{if } i = t(q) \text{ and } l = L+1\\ 0 & \text{otherwise}\end{cases}\\
& \qquad\qquad i\in\mathsf{V}^q,\ l\in\{2,\dots,L+1\},\ q\in\mathsf{Q} && (3.9)\\
& y \text{ binary} && (3.10)\\
& x \ge 0 \text{ and integer.} && (3.11)
\end{aligned}
$$

Each variable y_e states whether edge $e \in \mathsf{E}$ is selected and each variable x_{ij}^{lq} describes the amount of flow through arc (i, j, l) for commodity q in layered graph G^q. Constraints (3.9) are the flow conservation constraints at every node of the layered graph which guarantee that K units of flow go from $s(q)$ to $t(q)$, while constraints (3.8) guarantee edge-disjointness of the paths. Note that (3.8) together with (3.10) imply that $x_{ij}^{lq} \le 1$ for $i \ne j$, while (3.9) implies that $x_{dd}^{lq} \le K$.

3.3 Benders decomposition

This section reminds the general principle of Benders decomposition. Then it applies the framework to reformulate (**ML**) and (**HOP**).

3.3.1 General scheme

When facing a complex mixed-integer optimization problem, the Benders decomposition method (Benders, 1962) can be used to project out complicating real variables. This projection results in the addition of many additional constraints to the problem. Benders decomposition has been widely studied for fixed charge network design problems (Costa, 2005). Indeed, these problems usually route multi-commodity flows on some network to be designed. Therefore, the associated formulations contain many constraints and variables bound together by the capacity constraints. Then, once we project out the flow variables, the subproblems become independent linear programs for each commodity (see, for instance, **SHOP**$^q(\overline{y})$ below), thus reducing significantly the size of the linear programs to solve. For problem (**ML**), we can actually re-aggregate the different subproblems into a single LP with limited size. This is the well known capacitated formulation for network design problem (**ND**), where Benders cuts are replaced by metric inequalities, see Costa et al. (2009) for a deeper comparison of Benders cuts and metric inequalities for (**ND**). Because

(**ML**) has two layers, we have therefore two subproblems to solve, each one looking for violated metric inequalities in the corresponding layer, see Section 3.3.2.

Note that the classical framework does not apply to model (**HOP**) because all of its variables are integer; classical duality theory does not allow us to project out variables with integer restrictions. It is well known indeed in the field of stochastic programming that integer recourse cannot be tackled through classical Benders decomposition, called L-shaped in stochastic programming (Birge and Louveaux, 2008). Although Carøe and Tind (1998) generalize the L-shape to integer recourse using general duality theory, their framework stays mainly theoretical. To avoid this difficulty, we introduce a new formulation for (**HOP**), (**HOPc**), where we relax the integrality restrictions on x variables in (**HOP**), replacing (3.11) by

$$x \geq 0.$$

The question whether (**HOPc**) is equivalent to (**HOP**) is not trivial because (3.8) prevents (**HOP**) from being totally unimodular. It is answered partly in the following proposition (see Botton (2010); Fortz et al. (2010) for details):

Proposition 3.1. *The problems* (**HOP**) *and* (**HOPc**) *are equivalent for* $L = 2, 3$ *with any* $K \geq 2$*, and* $L = 4$ *with* $K = 2$.

Moreover, for all our test instances (see Section 3.5.2) we know from computational results that equivalence from Proposition 3.1 holds. Therefore, we apply Benders decomposition to (**HOPc**) in the sequel.

Benders decomposition aims at partitioning and delaying constraint generation. Consider the following mixed integer program

$$(\mathbf{P}) \qquad \begin{aligned} \min \quad & c^t x + k^t y \\ \text{s.t.} \quad & Ax + By \geq e && (3.12) \\ & Dy \geq f && (3.13) \\ & x \geq 0, \; y \geq 0 \text{ and integer.} \end{aligned}$$

For instance, given a network design problem, variables x and y represent flows and capacities, respectively. Constraints (3.12) contain all restrictions about routing and capacities installed on edges, whereas (3.13) describe upper and lower bounds on capacities.

We want to project variables x out of (**P**). For network design problems, this consists in describing the polyhedron containing feasible capacities y without explicitly describing the routing. In this purpose, let us express the (**P**) as

$$\min_{y \in \mathsf{Y}} \{k^t y + \min_{x \geq 0} \{c^t x \; : \; Ax \geq e - By\}\}, \tag{3.14}$$

where the set $\mathsf{Y} = \{y \mid Dy \geq f, \; y \geq 0 \text{ and integer}\}$ describes the admissible capacities for (**P**).

Because the subproblem in (3.14) is a linear program we can substitute it with its dual

$$\mathbf{SP}(y) = \quad \max_{s\geq 0}\{s^t(e-By) \ : \ s^tA \leq c\}.$$

Therefore, (3.14) becomes

$$\min_{y\in \mathsf{Y}}\{ky + \mathbf{SP}(y)\}. \tag{3.15}$$

Remark that the feasibility polyhedron of the subproblem does not depend on y. Let $\mathcal{P} = \{s \mid s \geq 0;\ s^tA \leq c\}$ represents this polyhedron. We assume that $\mathcal{P}$ is nonempty otherwise the former problem (3.14) was either infeasible or unbounded. $\mathcal{P}$ is therefore composed of extreme points s^p (for $p = 1,\dots,P$) and extreme rays r^q (for $q = 1,\dots,Q$). The solution of $\mathbf{SP}(y)$ is either bounded or unbounded. In the first case the solution is one of the extreme points s^p, $p = 1,\dots,P$. In the latter situation, there is a direction r^q for which $(r^q)^t(e - By) > 0$. The unbounded situation results in a infeasible primal problem and must be avoided. Hence subproblem $\mathbf{SP}(y)$ can be replaced by two set of inequalities for y, and (3.15) becomes the subsequent Master Problem

$$(\mathbf{MP}) \qquad \begin{array}{lll} \min & k^ty + \gamma & \\ \text{s.t.} & \gamma \geq s^p(e-By) & p = 1,\dots,P \qquad (3.16) \\ & (r^q)^t(e-By) \leq 0 & q = 1,\dots,Q \qquad (3.17) \\ & Dy \geq f & \\ & \gamma \geq 0,\ y \geq 0 \text{ and integer.} & \end{array}$$

The auxiliary variable γ and equations (3.16) replace the subproblem objective value, whereas equations (3.17) ensure that extreme rays are avoided.

Since there is usually a very large number of constraints in (**MP**), we should rather generate them dynamically during the solution algorithm. For instance, given a vector $\overline{y}$ and a real $\overline{\gamma}$ satisfying (3.16) for $p = 1,\dots,P' < P$ and (3.17) for $q = 1,\dots,Q' < Q$, we solve $\mathbf{SP}(\overline{y})$. If the problem is unbounded, we choose any unbounded extreme ray $\overline{r}$ and add a the feasibility cut $\overline{r}^t(e - By) \leq 0$ to (**MP**). Otherwise, let $\overline{s}$ be the optimal solution. If $\mathbf{SP}(\overline{y}) > \overline{\gamma}$, we add the optimality cut $\overline{s}^t(e - By) \leq 0$ to (**MP**), otherwise $(\overline{y}, \overline{\gamma})$ satisfies all constraints (3.16) and (3.17).

Notice that we have described only a general scheme, whose specific implementation may significantly impact the algorithm efficiency (Fischetti et al., 2010). Since neither of the objective functions of (**ML**) and (**HOP**) have costs associated with routing variables, we focus on the generation of feasibility cuts in what follows. In particular, we choose violated extreme rays by solving $\mathbf{SP}(\overline{y})$ augmented by a normalization constraint (see Fischetti et al. (2010); Ljubic et al. (2009) for testing of different normalization constraints). If its solution is strictly less than 0, we add a feasibility cut; otherwise no cut is added. The next subsections detail the subproblems used to look for violated feasibility cuts, while we present in Section 3.4 different algorithms to solve (**MP**) based on delayed feasibility constraint generation.

3.3.2 Reformulation of network design problems

We turn now to reformulations for (**MP**) and (**HOP**).

Multi-layer network design

For problem (**ML**), (**MP**) becomes the so-called capacitated formulation (Knippel and Lardeux, 2007).

$$(\mathbf{MML}) \qquad \begin{array}{ll} \min & \sum_{e\in \mathsf{E}^l} c^l_e y^l_e + \sum_{f\in \mathsf{E}^u} c^u_f y^u_f \\ \text{s.t.} & y^l \in \mathsf{Y}^l(y^u) \\ & y^u \in \mathsf{Y}^u, \end{array}$$

where sets $\mathsf{Y}^l(y^u)$ and Y^u are defined by metric inequalities:

$$\mathsf{Y}^l(y^u) = \left\{ y \in \mathbb{Z}_+^{|\mathsf{E}^l|} \mid \forall \lambda \in \mathcal{M}_{|\mathsf{V}|},\, C^l \sum_{ij\in \mathsf{E}^l} \lambda_{ij} y^l_{ij} \geq C^u \sum_{ij \in \mathsf{E}^u} \lambda_{ij} y^u_{ij} \right\} \tag{3.18}$$

and

$$\mathsf{Y}^u = \left\{ y \in \mathbb{Z}_+^{|\mathsf{E}^u|} \mid \forall \lambda \in \mathcal{M}_{|\mathsf{V}|},\, C^u \sum_{ij\in \mathsf{E}^u} \lambda_{ij} y^u_{ij} \geq \sum_{i<j} \lambda_{ij} d_{ij} \right\}, \tag{3.19}$$

where the metric cone (Deza and Laurent, 1997) $\mathcal{M}_n$ is defined by

$$\mathcal{M}_n = \{\lambda \in \mathbb{R}^{n(n-1)/2} \mid \lambda_{ij} \leq \lambda_{il} + \lambda_{lj}, \forall 1 \leq i < j \leq n,\ \forall 1 \leq l \leq n,\ j \neq l \neq i\}.$$

Given a capacity vector $(\overline{y}^l, \overline{y}^u)$, we can test its feasibility by solving the separation LP's $\mathbf{SML}(C^u\overline{y}^u, d)$ and $\mathbf{SML}(C^l\overline{y}^l, C^u\overline{y}^u)$, with $\mathbf{SML}(z,t)$ defined by

$$\mathbf{SML}(z,t) = \qquad \begin{array}{ll} \min & \sum_{i<j} \lambda_{ij} z_{ij} - \sum_{i<j} \lambda_{ij} t_{ij} \\ \text{s.t.} & \sum_{1\leq i<j\leq n} \lambda_{ij} = 1 \\ & \lambda \in \mathcal{M}_{|\mathsf{V}|} \end{array} \tag{3.20}$$

for any vectors $z, t \in \mathbb{R}^{n(n-1)/2}$. Normalization constraint (3.20) bounds the LP. If $\mathbf{SML}(z,t) < 0$, the solution $\overline{\lambda}$ leads to a metric inequality violated by (z,t):

$$\sum_{i<j} \overline{\lambda}_{ij} z_{ij} - \sum_{i<j} \overline{\lambda}_{ij} t_{ij} \geq 0. \tag{3.21}$$

Metric inequalities (3.18) and (3.19) are weak when capacities are modular. A simple way of strengthening the inequalities in (3.19) without increasing the complexity of the separation algorithm is to round coefficients of

$$\sum_{ij\in \mathsf{E}^u} C^u \lambda_{ij} y^u_{ij} \geq \sum_{i<j} \lambda_{ij} d_{ij} := d. \tag{3.22}$$

If $C^u\lambda_{ij}$ is integer for each $ij = f \in \mathsf{E}^u$, let $\gcd(C^u\lambda)$ be the greatest common divisor of those integers. Hence, dividing both sides of (3.22) by $\gcd(C^u\lambda)$ and rounding up $d/\gcd(C^u\lambda)$, we get the stronger cut

$$\sum_{ij\in\mathsf{E}^u} \frac{C^u\lambda_{ij}}{\gcd(C^u\lambda)} y_f^u \geq \left\lceil \frac{d}{\gcd(C^u\lambda)} \right\rceil. \tag{3.23}$$

We show in Section 3.5 the effect of these stronger cuts.

Note that Avella et al. (2007) introduced the *Tight Metric Inequalities*, which completely describe Y^u. However, since they are $\mathcal{NP}$-hard to separate, we do not consider them in this thesis.

Hop-constrained path diversified network design

Projecting out x variables from (**HOPc**), we obtain the subsequent Master Problem

$$(\mathbf{MHOP}) \qquad \begin{array}{lll} \min & \displaystyle\sum_{e\in\mathsf{E}^l} c_e y_e & \\ \text{s.t.} & K\overline{\pi}_{t(q)}^{L+1} - \displaystyle\sum_{e\in\mathsf{E}^l} y_e\overline{\sigma}_e \leq 0 & (\overline{\pi},\overline{\sigma}) \in \displaystyle\bigcup_{q\in\mathsf{Q}} \mathsf{R}^q \\ & y_e \in \{0,1\}, & \end{array}$$

where R^q contains extreme rays (vertices belonging to the normalization hyperplane, in fact) of the feasibility polyhedron for the dual subproblem $\mathbf{SHOP}^q(\overline{y})$ described next. Given commodity $q \in \mathsf{Q}$, let us introduce a dual variable π_i^l, associated with node $i \in \mathsf{V}$ and layer l, for each constraint (3.9) and a dual variable σ_e for each constraint (3.8). Defining $o := s(q)$ and $d := t(q)$, and adding the constraint $\pi_d^{L+1} \leq 1$ to normalize the dual cone, we get the dual subproblem

$$\begin{array}{rlll} \mathbf{SHOP}^q(\overline{y}) = & \max & K\pi_d^{L+1} - \displaystyle\sum_{ij\in\mathsf{E}} \overline{y}_{ij}\sigma_{ij} & \\ & \text{s.t.} & \pi_i^2 - \pi_o^1 - \sigma_{oi} \leq 0 & oi \in \mathsf{E} \\ & & \pi_i^{l+1} - \pi_j^l - \sigma_{ij} \leq 0 & ij \in \mathsf{E}, i,j \notin \{o,d\}, l \in \{2,\dots,L\} \\ & & \pi_j^{l+1} - \pi_i^l - \sigma_{ij} \leq 0 & ij \in \mathsf{E}, i,j \notin \{o,d\}, l \in \{2,\dots,L\} \\ & & \pi_d^{L+1} - \pi_i^L - \sigma_{id} \leq 0 & id \in \mathsf{E}, l \in \{2,\dots,L\} \\ & & \pi_d^{l+1} - \pi_d^l \leq 0 & l \in \{2,\dots,L\} \\ & & \pi_d^{L+1} \leq 1 & \\ & & \sigma_{ij} \geq 0 & ij \in \mathsf{E}. \end{array}$$

Note that for each commodity $q \in \mathsf{Q}$, one of the constraints in (3.9) is redundant, which can be represented by setting $\pi_o^1 = 0$.

3.4 Algorithms

In this section, we consider the general master problem (**MP**) and a set of subproblems $\mathbf{SP}^q(\overline{y})$, $q \in \mathsf{Q}$, yielding Benders cut

$$\overline{r}^t(e - By) \leq 0, \tag{3.24}$$

and that integer vector of variables y has $|\mathsf{E}|$ components. Note that decomposing (**ML**) leads to only two (or three, see `sc` below) different subproblems. The following description is valid for both (**ML**) and (**HOP**), apart from algorithms `sc` and `mc` tailored for (**ML**), and `heuristic` tailored for (**HOP**). Problem (**MP**) contains an exponential number of constraints while only a few of them are active at the optimum. Therefore, we must dynamically generate the required constraints throughout the solution method. First works on Benders decomposition for mixed-integer problems use cutting plane algorithms, cycling many times between master integer problems and continuous subproblems. However, modern developments in branch-and-cut frameworks such as the commercial CPLEX (IBM-ILOG, 2009) and the noncommercial SCIP (Achterberg, 2009), among others, have eased the development of a branch-and-cut algorithm to solve the master problem, incorporating the Benders cut separation in the cutting plane callback. In subsection 3.4.1 we briefly describe the multi-cut cutting plane algorithm, while we detail our different branch-and-cut algorithms in subsection 3.4.2.

3.4.1 Cutting plane approach

Algorithm 1: "Naive" Benders decomposition algorithm: `cp`

```
repeat
    solve (MP);                                              /* solve an IP */
    let $\overline{y}$ be an optimal solution;
    foreach $q \in \mathsf{Q}$ do
        compute $\mathbf{SP}^q(\overline{y})$;                    /* solve the dual subproblem */
        if $\mathbf{SP}^q(\overline{y}) > 0$ then add (3.24) to (MP);
until $s_q \leq 0$ for each $q \in \mathsf{Q}$;
return $\overline{y}$
```

In Algorithms 1 and 2, we describe our cutting plane algorithms, `cp` and `cp-i`. Because the main computational burden is the solution of (**MP**), we implemented a multi-cut version of the algorithm: we solve the subproblem for each commodity, therefore adding up to $|\mathsf{Q}|$ cuts per iteration. The improved version `cp-i` starts by solving the linear programming relaxation of (**MP**) in a cutting plane fashion. Various works enhance this classical solution algorithm. Among them, Magnanti and Wong (1981) study the effect of using special cuts called "pareto optimal", Tsamasphyrou et al. (2000) describe a more subtile version of the multi-cut algorithm,

Algorithm 2: Improved Benders decomposition algorithm: `cp-i`

```
repeat
    solve the linear programming relaxation of (MP);          /* solve a LP */
    let $\overline{y}$ be an optimal solution;
    foreach $q \in \mathsf{Q}$ do
        compute $\mathbf{SP}^q(\overline{y})$;
        if $\mathbf{SP}^q(\overline{y}) > 0$ then add (3.24) to (MP);
until $s_q \leq 0$ for each $q \in \mathsf{Q}$;
repeat
    solve (MP);                                                /* solve an IP */
    let $\overline{y}$ be an optimal solution;
    foreach $q \in \mathsf{Q}$ do
        compute $\mathbf{SP}^q(\overline{y})$;
        if $\mathbf{SP}^q(\overline{y}) > 0$ then add (3.24) to (MP);
until $s_q \leq 0$ for each $q \in \mathsf{Q}$;
return $\overline{y}$
```

grouping together subsets of subproblems, and Rei et al. (2009) use local branching to accelerate the overall algorithm. Sometimes, additional classes of cuts are known for the problem allowing to add even more cuts at each iteration. For instance, considering (**ML**), Gabrel et al. (1999) describe the following cutting plane algorithms.

- `sc` Their *single constraint generation* adds up to three cuts per iteration. Besides cuts coming from $\mathbf{SML}(C^u\overline{y}^u, d)$ and $\mathbf{SML}(C^l\overline{y}^l, C^u\overline{y}^u)$, they also consider cuts from subproblem $\mathbf{SML}(C^l\overline{y}^l, d)$:

$$C^l \sum_{i<j} \overline{\lambda}_{ij} y^l_{ij} - \sum_{i<j} \overline{\lambda}_{ij} d_{ij} \geq 0. \tag{3.25}$$

  Although cuts (3.25) are not needed to ensure feasibility, they help to reduce the number of required iterations by forcing y^l to take sensible values, especially in the first few iterations.

- `mc` Aiming to reduce the iterations of Algorithm 1 even further, they introduce *multiple constraint generation.* This algorithm adds the same cuts as `sc` plus a few bipartition inequalities violated by $(\overline{y}^l, \overline{y}^u)$, at each iteration.

Another situation occurs when the subproblems are separable, i.e., they can be decomposed into several problems, the solution of which results in the addition of a cut to (**MP**). See, for example, the multi-cut L-shaped algorithm for stochastic programming problems with recourse (Birge and Louveaux, 2008).

3.4.2 Branch-and-cut approach

An alternative strategy is to solve (**MP**) only once. We aim at embedding the generation of violated feasibility cuts (3.24) into the branch-and-cut framework solving (**MP**).

It is important to add many cuts early in the tree to avoid exploration of too many infeasible nodes. However, adding too many unnecessary cuts would slow down the linear programming relaxation at each node. Our first branch-and-cut algorithm, `bc-all`, checks for violated Benders cuts (3.24) at every node of the tree. However, this algorithm is relatively slow, because too many cuts are added making (**MP**) unnecessarily large and too much time is spent in the solution of $\mathbf{SP}^q(\overline{y})$. In `bc-int`, we check for cuts (3.24) only at the root and at integer nodes. Finally, we developed a hybrid algorithm `bc-n`, described in Algorithm 3, checking for violated inequality (3.24) at integer nodes and nodes with a depth less than or equal to n. Note that `bc-n` generalizes both frameworks since `bc-int` is the same as `bc-0`, and `bc-all` is the same as `bc-`$|\mathsf{E}|$.

In Algorithm 3, solving a node $o' \in \mathsf{T}$ means solving the linear programming relaxation of (**MP**), augmented with branching constraints of o', while depth(o') counts the number of branching constraints of o'.

3.4.3 Heuristic

An intrinsic difficulty of Benders decomposition is that we replace the well-structured problem (**P**) by a problem (**MP**) with no straightforward structure. Indeed, it is well known that special structures can help the solution of hard integer programs. For instance, detecting a a flow structure within a more complicated problem can be used to add strong cut inequalities (Achterberg and Raack, 2010). Moreover, for many problems Benders cuts have fractional coefficients, yielding numerical instability (Codato and Fischetti (2006) avoid this difficulty for combinatorial problems with certain classes of "big M" constraints). Finally, it will be hard for our default MIP solver (CPLEX 11 in our case) to find good upper bounds. We present next a simple, yet efficient, heuristic for (**HOP**). First, we solve the linear programming relaxation of the Benders decomposition, resulting in a fractional $\overline{y}$. Then, for each $\overline{y}_e = 0$ we add the constraint $y_e = 0$ to (**MHOP**), and we solve the resulting problem with `bc-n`. This allows us to reduce significantly the number of variables of the problem, yielding a very good solution in a limited amount of time. The issue whether or not the heuristic finds always a feasible solution is discussed in (Botton, 2010; Fortz et al., 2010).

3.5 Computational Experiments

The role of this section is two-folds. First, we want to underline the improvement gained by the use of branch-and-cut algorithms instead of cutting planes algorithms,

Algorithm 3: Hybrid branch-and-cut algorithm: `bc-n`

```
begin                                                        /* Initialization */
    $\mathsf{T} = \{o\}$ where $o$ has no branching constraints;
    $UB = +\infty$;
while $\mathsf{T}$ is nonempty do
    select a node $o' \in \mathsf{T}$;
    $\mathsf{T} \leftarrow \mathsf{T} \backslash \{o'\}$;                     /* withdraw node $o'$ from the tree */
    solve $o'$;
    let $\overline{y}$ be an optimal solution;
    let $\overline{w}$ be the optimal cost;
    if $\overline{w} < UB$ then
        if $\overline{y} \notin \mathbb{Z}^{|\mathsf{E}|}$ and $depth(o') \geq n+1$ then
            branch, resulting in nodes $o^*$ and $o^{**}$;
            $\mathsf{T} \leftarrow \mathsf{T} \cup \{o^*, o^{**}\}$;                /* add children to the tree */
        else
            foreach $q \in \mathsf{Q}$ do compute $\mathbf{SP}^q(\overline{y})$;
            if $\mathbf{SP}^q(\overline{y}) > 0$ then add (3.24) to (MP);
            if $\mathbf{SP}^q(\overline{y}) > 0$ for some $q \in \mathsf{Q}$ then
                $\mathsf{T} \leftarrow \mathsf{T} \cup \{o'\}$;              /* put node $o'$ back in the tree */
        if $\overline{y} \in \mathbb{Z}^{|\mathsf{E}|}$ and $\mathbf{SP}^q(\overline{y}) \leq 0$ for each $q \in \mathsf{Q}$ then
            $UB \leftarrow \overline{w}$;                             /* define a new upper bound */
            $y^* \leftarrow \overline{y}$;                              /* save current incumbent */
return $y^*$
```

even when additional cutting planes are generated. This is realized by comparing `bc-int`, `sc`, and `mc` on various instances for problem (**ML**), see Section 3.5.3. We also compare these algorithms with `extended` and test the effect of rounding cuts. Then, we further investigate the impact of correctly implementing branch-and-cut algorithms. Namely, we test various ways of implementing cut generation as well as the effect of using a good heuristic for problem (**HOP**). We present in Section 3.5.4 comparative results of algorithms `extended`, `cp`, `cp-i`, `bc-all`, `bc-int`, `bc-n`, and `bc-n-heur`.

In the next section we detail how we have implemented all algorithms, while instances are described in Section 3.5.2.

3.5.1 Implementation details

All models have been coded in JAVA using CPLEX 11 MIP solver. Algorithms for (**ML**) were run on a HP Compaq 6510b with an Intel Core 2 Duo processor at 2.40 GHz and 2 GB of RAM memory, while those for (**HOP**) were run on a DELL Latitude D820 with a processor Intel Core Duo 2 T7200 of 2GHz and 2.5 GB of RAM memory. We allow CPLEX to store the branch-and-bound tree in a file, setting parameter *IntParam.NodeFileInd* to 2, to avoid from running out of memory. Moreover, for each algorithm we configure CPLEX as follows :

`extended`: This algorithm solves the complete models (**ML**) and (**HOPc**). All parameters have been kept to their default values, CPLEX chooses to explore the branch-and-bound tree with the dynamic search.

`cp,sc,mc`: We build an empty IP for the master problem, and one LP for each Benders subproblem. Then, we cycle between these problems, with all parameters kept to their default values. CPLEX uses the dynamic search for solving all IP's.

`cp-i`: We build an empty LP for the master problem and solve it through a cutting plane algorithm. Then, we build an IP for the master problem, the constraints of which are the Benders cuts just generated; we solve it through a cutting plane algorithm, with all parameters kept to their default values. CPLEX uses the dynamic search.

`bc-all`, `bc-int`, and `bc-n`: Since the model does not contain explicitly all constraints, we must deactivate the dual presolve, setting *BooleanParam.PreInd* to false. Then, we implemented our (global) cuts generation with a *LazyConstraintCallback*, preventing CPLEX from using dynamic search.

`heuristic`: We first solve the linear programming relaxation by a cutting plane algorithm (in fact, we use a branch-and-cut algorithm with a limit of 0 nodes, setting *IntParam.NodeLim* to 0). Then, we fix some of the variables to 0, and re-solve the resulting problem with `bc-n`.

`bc-n-heur`: We use the algorithm `bc-n`, providing CPLEX with the upper bound found by `heuristic`. The CPU times reported do not consider the time spent in `heuristic`.

3.5.2 Instances

We describe below the different sets of instances used in our computational experiments.

Random

The first 35 instances for (**ML**) are randomly generated and share the next features: $C^u = 64$, $C^l = 128$, $\mathsf{G}^u = (\mathsf{V}, \mathsf{E}^u)$ is a complete graph. Demands are random integers uniformly generated between 0 and 64 for each pair of nodes and the cost of any edge $e \in \mathsf{E}^l \cup \mathsf{E}^u$ is based on the distance between the extremities of e.

TC and TE

The sets TC and TE for (**HOP**) were taken from a class of complete graphs $\mathsf{G} = (\mathsf{V}, \mathsf{E}^l)$, reported by Gouveia (1996). They share the following features: $|\mathsf{V}| = 21$, $|\mathsf{Q}| \in \{5, 10\}$, and all point-to-point demands share one of their extremities. The cost matrix for each instance considers the integer part of the Euclidean distance between the coordinates of the 21 nodes, randomly placed among the integer points of a grid 100×100. The TC class contains 5 instances with 5 commodities and 5 instances with 10 commodities with the root located in the center of the grid and the TE class contains 5 instances with 5 commodities and 5 instances with 10 commodities with the root located on a corner of the grid. We see in the next section that instances TE are much harder to solve than instances TC.

SNDlib

We consider also instances from the *SNDlib* repository (Orlowski et al., 2007). For (**ML**), these networks have been taken as lower layers; upper layers are complete graphs. Full details of these instances are shown in Table 3.1.

Name	$\lvert\mathsf{V}\rvert$	$\lvert\mathsf{E}^l\rvert$	$\lvert\mathsf{Q}\rvert$	(**ML**)	(**HOP**)
pdh	11	34	27	Yes	Yes
di-yuan	11	42	42	Yes	Yes
dfn-gwin	11	47	9	Yes	Yes
polska	12	18	17	Yes	Yes
nobel-us	14	21	33	Yes	Yes
atlanta	15	22	55	Yes	No

Table 3.1: Instances description.

3.5.3 Multi-layer network design

We fix a time limit of 3600 seconds for instances with 8 and 9 nodes. The corresponding Time/Gap column gives either the solution time in seconds or the gap when the time limit is reached. For instances 26–35 and the ones from SNDlib, we allow up to 18000 seconds, reporting the status after 3600 seconds. The reported solution times consider for the whole durations of the algorithms. Underlined gaps indicate memory overflows.

We can see in Tables 3.2 and 3.3 that `sc`, `mc` and `bc-int` outperform `extended` by far. `bc-int` is always faster than both `sc` and `mc`. The ratio between the solution time of `bc-int` and the one of the faster cutting plane algorithm ranges from 2.2 to 34.4 with a geometric average of 10.7. This is explained by the high number of iterations performed by both cutting plane algorithms, where each iteration is required to solve an IP to optimality. However, the ratio is still far from the number of iterations, since many of the iterations contain only a few cuts.

`bc-int` usually generates more cuts than `sc`, even though `sc` generates cuts of type (3.25). Thus, many of these cuts are not needed to ensure the feasibility of the solution. Hence more efficient management of the cut pool, eliminating the non active cuts, may improve Algorithm 3.

The relative performance of `sc` and `mc` is as expected: `mc` adds many more cuts than `sc`, resulting in fewer iterations and shorter solution times. See (Knippel and Lardeux, 2007) for a more detailed comparison of `sc` and `mc`.

Note that the cutting plane algorithms were unable to solve any of the larger instances within 18000 seconds. Hence, in Tables 3.5, 3.6 and 3.7 we compare `extended` and `bc-int` with normal and rounded cuts ((3.22) and (3.23), respectively) for those instances, denoted by `nc` and `rc`, respectively. Although `nc` and `rc` beat `extended` for most instances, the difference is much smaller than it is for easier instances from Tables 3.2 and 3.3.

Results from Table 3.6 show that `extended` explores hundreds of thousands of nodes, whereas both `nc` and `rc` explore millions of them. Note that the number of nodes explored by `bc-int` grows rapidly with the problem size. `extended`, however, manages to compute good bounds for hard instances, while exploring a relatively small tree.

$\|E^l\|$	Time/Gap	Time			Cuts generated			Iterations		Explored nodes		Times ratio
	`extended`	`sc`	`mc`	`bc-int`	`sc`	`mc`	`bc-int`	`sc`	`mc`	`extended`	`bc-int`	$\frac{min(\texttt{sc},\texttt{mc})}{\texttt{bc-int}}$
14	433	32.7	13.1	0.7	76	155	84	52	38	110345	1911	18.7
14	*1.95%*	9.2	9.1	0.6	76	177	85	46	33	926934	1886	15.1
14	3508	29.9	31.8	4.1	101	168	115	48	32	668218	13259	7.3
14	387	33.1	27.5	0.8	78	202	84	45	37	83482	1989	34.4
14	183	4.9	4.2	0.2	63	144	71	34	23	37388	76	21
16	984	54.1	37.9	1.2	88	183	92	54	30	184589	3282	31.2
16	508	33.8	15.4	0.8	83	187	88	39	24	118575	2310	19.3
16	2434	21.2	34.4	4.9	86	186	89	63	45	659259	26834	4.3
16	856	104.2	66.6	3.0	121	189	118	58	29	118797	8862	22.2
16	3487	33.3	16.2	1.8	90	171	99	54	26	803442	6099	9

Table 3.2: Results of `extended`, `sc`, `mc` and `bc-int` on randomly generated instances with 8 nodes.

3.5.4 Hop-constrained path diversified network design

First, we look at the quality of the linear programming relaxation of our model (**HOPc**). Let IP^* and LP^* be the optimal value of (**HOPc**) and its linear programming relaxation, respectively. Table 3.8 shows that the gap $\left(\frac{IP^*-LR^*}{IP^*} * 100\right)$ decreases as K increases. Before comparing the different algorithms, we need to determine the "best" value for the depth parameter of branch-and-cut algorithm `bc-n`. We select a group of complicated instances (instances that `extended` can not solve to optimality within 3600 seconds) and we test different values of the depth

$\lvert E^l \rvert$	Time/Gap	Time			Cuts generated			Iterations		Explored nodes		Times ratio
	`extended`	`sc`	`mc`	`bc-int`	`sc`	`mc`	`bc-int`	`sc`	`mc`	`extended`	`bc-int`	$\frac{min(\texttt{sc},\texttt{mc})}{\texttt{bc-int}}$
16	2002	363.8	170.0	12.3	119	303	128	87	50	231606	42675	13.8
16	3063	217.8	244.5	15.5	115	310	186	68	43	284648	36745	14.1
16	538	226.2	264.0	14.3	149	272	156	95	39	58767	38257	15.8
16	*4.16%*	1639.2	450.1	25.3	127	275	156	85	46	232556	74618	17.8
16	*0.72%*	125.5	67.6	4.7	112	315	104	73	42	318587	14759	14.4
18	*2.93%*	190.5	143.3	66.1	149	328	164	103	47	420776	241323	2.2
18	*3.94%*	529.6	272.2	14.8	129	299	147	72	40	340148	39818	18.4
18	*1.54%*	109.3	55.2	8.4	111	261	154	66	45	293697	19807	6.6
18	539	60.0	21.9	0.9	92	225	114	56	28	55293	1461	24.3
18	*0.63%*	425.6	224.1	78.0	160	286	192	79	41	265740	143706	2.9
20	*1.96%*	67.2	53.3	13.7	100	261	105	60	40	313350	58887	3.9
20	*3.35%*	415.0	201.2	62.8	131	341	165	83	48	282078	209571	3.2
20	*6.07%*	293.8	67.7	28.3	130	266	155	83	36	283205	91849	2.4
20	*3.1%*	–	730.7	66.4	–	290	187	–	47	266132	174850	11.0
20	*2.32%*	193.2	217.3	12.0	113	227	121	72	41	371485	44940	16.1

Table 3.3: Results of `extended`, `sc`, `mc` and `bc-int` on randomly generated instances with 9 nodes.

$\lvert E^l \rvert$	Initial cuts		3600 seconds		18000 seconds	
	`nc`	`rc`	`nc`	`rc`	`nc`	`rc`
20	111	145	197	129	67	–
20	105	145	265	240	12	–
20	142	204	305	266	40	19
20	105	162	373	322	14	6
20	124	165	267	120	–	–
25	144	161	544	368	30	37
25	127	222	398	332	7	22
25	131	165	231	242	8	11
25	122	179	391	216	7	–
25	129	187	347	225	37	70

Table 3.4: Number of cuts generated by `bc-int` with normal and rounded cuts (`nc` and `rc` respectively) at different steps of Algorithm 3, on randomly generated instances with 10 nodes.

parameter n. On Figure 3.2, we plot the result of this tuning stage. For both curves, the minimum is reached when $n = 5$. Therefore, in the sequel we always consider `bc-5` for the hybrid branch-and-cut algorithm.

We compare the performance in terms of resolution time for the different methods by plotting the performance profile (Dolan and More, 2002) on Figure 3.3. Clearly, algorithms `bc-5`, `bc-int` and `bc-5-heur` are the fastest algorithms. Moreover, we see that the simple and naive implementation of the Benders decomposition `cp` performs much worse than the original model `extended`. Indeed, Benders decomposition suffers from the loss of problem structure, so that each of the iterations of the master problem requires a sensible amount of time (see Table 3.12 for aver-

$\lvert E^l \rvert$	Time/Gap (3600 seconds)			Time/Gap (18000 seconds)		
	`extended`	`nc`	`rc`	`extended`	`nc`	`rc`
20	*3.43%*	*0.53%*	745	*2.71%*	4109	–
20	*4.33%*	*1.22%*	1965.7	*2.2%*	77801	–
20	*4.88%*	*5.09%*	*2.86%*	*4.06%*	*4.72%*	*0.55%*
20	*1.19%*	*4.23%*	*1.37%*	7802	*2.77%*	9355
20	*2.76%*	368.9	79	*1.41%*	–	–
25	*6.43%*	*3.66%*	*4.54%*	*3.96%*	*2.47%*	*4.1%*
25	*2.82%*	*2.69%*	*0.79%*	*2.03%*	*2.37%*	5032
25	*5.13%*	*0.65%*	*1.09%*	*3.98%*	4768	5922
25	*1.16%*	*1.05 %*	1447	8301	5076	–
25	*3.3%*	*2.5%*	*1.37%*	*2.35%*	*1.98%*	9963

Table 3.5: Solution times for `extended`, `nc` and `rc` at different steps of Algorithm 3, on randomly generated instances with 10 nodes.

$\lvert E^l \rvert$	3600 seconds			18000 seconds		
	`extended`	`nc`	`rc`	`extended`	`nc`	`rc`
20	148455	5352019	1214254	747087	6377687	–
20	139125	5722520	2716054	714667	11484670	–
20	122756	2846870	23400446	648448	5337499	13068541
20	144445	284670	2866228	375985	5337499	8042525
20	148609	408489	103577	820049	–	–
25	105340	3112489	3618044	575839	5179420	5469716
25	115136	3471580	3174756	657367	5552061	4553309
25	125850	3917977	4222550	701442	5563211	7104511
25	150276	2676776	1248257	404154	3979981	–
25	147617	3514051	4155845	783877	6664584	11265256

Table 3.6: Number of explored nodes by `extended`, `nc` and `rc`, on randomly generated instances with 10 nodes.

ages numbers of iterations). Thus, a careful implementation is required to make the decomposition efficient.

Table 3.9 indicates, for each algorithm, the number of instances (out of the 213 instances which compose the entire test set) that can not be solved within the 3600 seconds. Among the 12 instances that `extended` can not solve to optimality, only 1 can not be solved by `bc-5`. `bc-5-heur` can solve all instances to optimality. The reader can find the arithmetic averages of CPU times on Table 3.10 (> indicates that one or more instances could not be solved to optimality).

Table 3.5.4 indicates the arithmetic average values of LP relaxation and `heuristic` gaps, given by $\frac{IP^*-LP^*}{IP^*}$ and $\frac{heuristic^*-IP^*}{IP^*}$, respectively, as well as `heuristic` CPU

Instances	T/G (3600 seconds)			T/G (18000 seconds)		
	`extended`	`nc`	`rc`	`extended`	`nc`	`rc`
pdh	*3.07%*	*0.84%*	3343.4	*2.53%*	7230	–
di-yuan	*1.38%*	*1.87%*	*1.9%*	10450	8710	9586
dfn-gwin	*3.77%*	*1.13%*	*1.18%*	*3.36%*	*0.54%*	*0.98%*
polska	*3.66%*	*0.32%*	*0.47%*	*2.45%*	*0.32%*	*0.23%*
nobel-us	*8.02%*	*8.01%*	*6.79%*	*3.92%*	*1.03%*	*1.51%*
atlanta	*3.92%*	314	707.9	*0.09%*	–	–

Table 3.7: Solution times for `extended`, `nc` and `rc`on instances based on networks from SNDlib.

L/K	1	2	3
3	21.40	10.90	5.47
4	24.30	9.45	5.69
5	26.70	6.00	5.16

Table 3.8: Geometric average of gap $\left(\frac{IP^*-LR^*}{IP^*} * 100\right)$ for all instances.

time in seconds and the number of instances for which the solution of `heuristic`is optimal. It can be seen that `heuristic` always provide a very good solution to the problem. Furthermore, `heuristic` is also pretty fast, taking around 4 seconds whereas `bc-5-heur` and `extended` take, respectively, on average 74.03 and more than 582.05 seconds. In 139 cases out of 213 (around 65%), the solution given by the heuristic is the optimal one. Finally, Table 3.13 and 3.14 present arithmetic averages of the number of Benders cuts generated by the branch-and-cut algorithms, and number of nodes explored by branch-and-cut and extended formulations, respectively, and Table 3.15 provides means of CPU time spent for solving Benders subproblems and the corresponding fraction in the total CPU time (means have been taken over all instances).

`extended`	`cp`	`cp-i`	`bc-all`	`bc-int`	`bc-5`	`bc-5-heur`
12	90	30	20	4	1	0

Table 3.9: Number of instances (out of 213) unsolved within one hour.

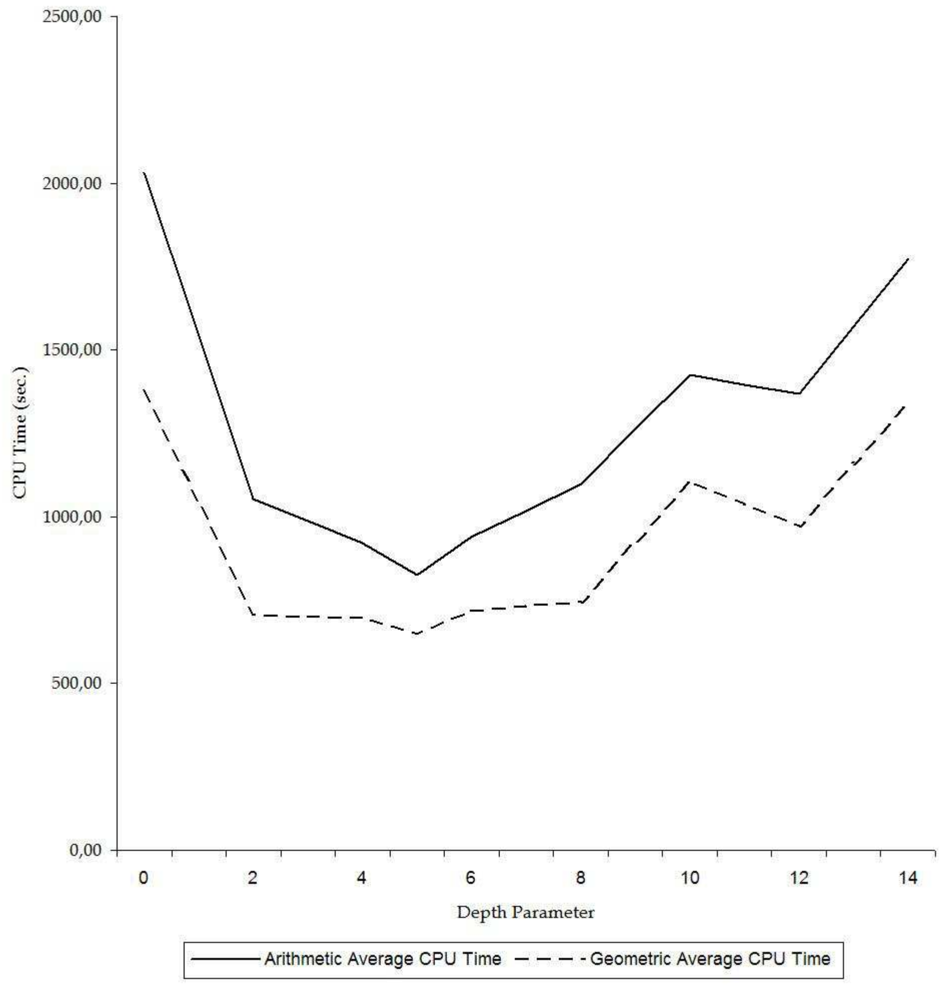


Figure 3.2: `bc-n` depth parameter tuning by average CPU time (sec.)

`extended`	`cp`	`cp-i`	`bc-all`	`bc-int`	`bc-5`	`bc-5-heur`	`heuristic`
> 498.09	> 1817.87	> 680.27	> 494.40	> 151.87	> 89.78	74.03	3.78

Table 3.10: Average CPU times for all approaches.

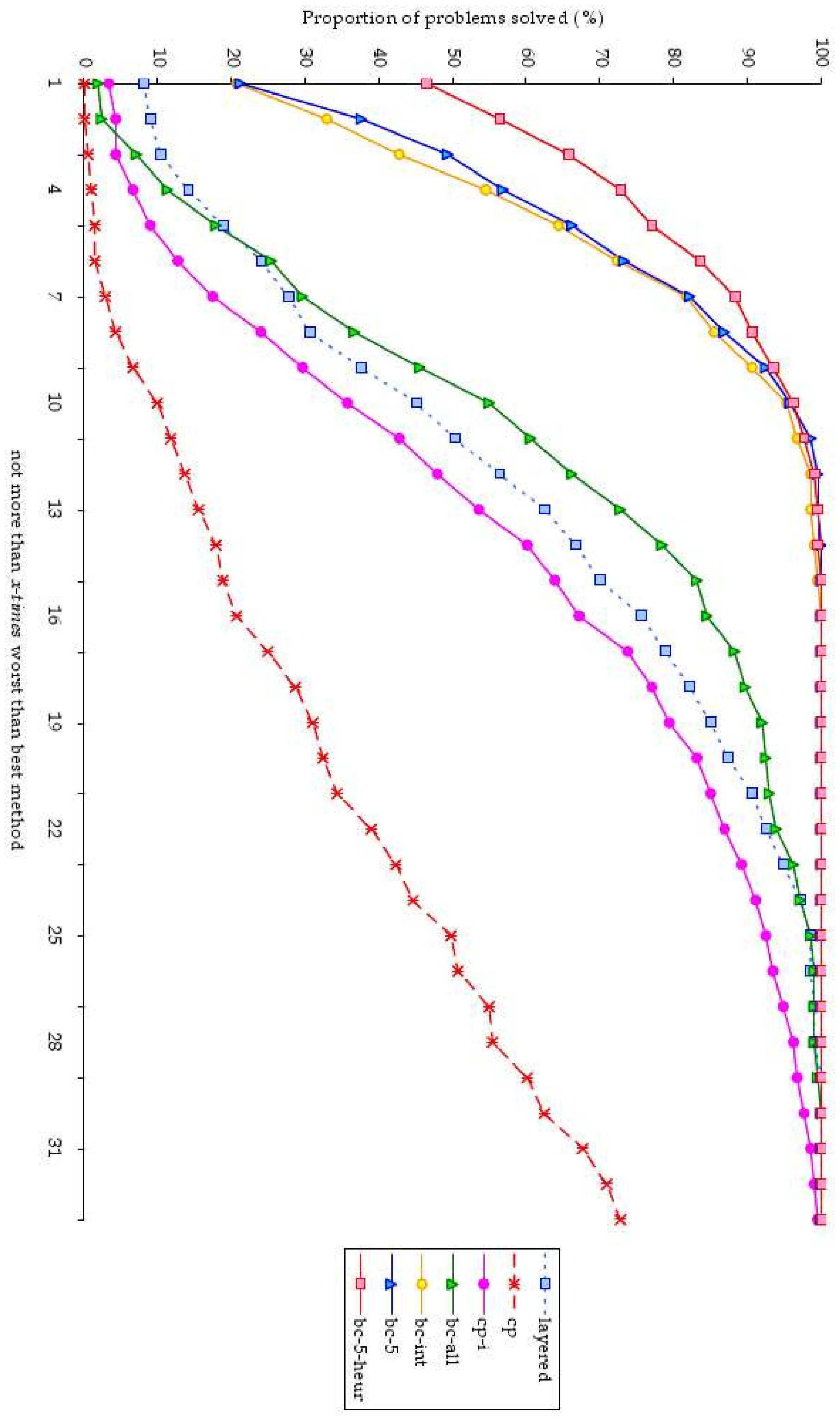


Figure 3.3: Performance profile comparing methods on the entire test set.

Instances	LP Relaxation Gap(%)	`heuristic` Gap(%)	Optimal	heuristic CPU Time
TC-5	8.04	1.77	24/45	0.76
TC-10	12.96	0.42	31/45	4.78
TE-5	10.67	2.17	24/45	0.97
TE-10	17.72	0.36	30/45	9.39
pdh	21.25	0.00	9/9	6.51
di-yuan	20.96	0.00	9/9	2.21
dfn-gwin	16.33	0.41	3/6	0.59
polska	11.11	0.00	5/5	0.37
nobel-us	10.00	0.00	4/4	1.39
Arithmetic Mean	13.13	1.01	-	3.78

Table 3.11: Linear relaxation gap and `heuristic` performance for the entire test set.

Instances	`cp`	`cp-i` linear (**MHOP**)	integer (**MHOP**)
TC-5	54.07	20.53	11.11
TC-10	97.00	28.22	22.09
TE-5	229.87	34.71	36.64
TE-10	126.29	37.58	51.51
pdh	174.67	13.33	111.44
di-yuan	81.33	9.89	20.67
dfn-gwin	94.33	17.00	29.50
polska	56.60	11.40	4.80
nobel-us	147.25	11.50	21.75
Arithmetic Mean	124.73	27.52	32.57

Table 3.12: Average numbers of iterations.

Instances	Number of cuts generated			
	bc-all	bc-int	bc-5	bc-5-heur
TC-5	342.98	120.33	128.20	94.71
TC-10	6616.56	567.80	780.49	378.36
TE-5	757.13	236.62	248.40	159.73
TE-10	17154.53	1693.60	2008.31	1029.40
pdh	6270.33	1219.22	1205.89	1103.22
di-yuan	1349.44	585.22	592.22	541.56
dfn-gwin	1221.33	289.67	310.17	200.83
polska	110.00	110.20	103.80	105.00
nobel-us	224.25	238.75	219.25	234.75
Arithmetic Mean	5617.64	644.65	760.01	433.20

Table 3.13: Number of cuts generated for the entire test set.

Instances	Number of nodes visited				
	bc-all	bc-int	bc-5	bc-5-heur	extended
TC-5	135.27	405.69	311.98	917.58	149.09
TC-10	885.60	6825.44	3545.40	13972.93	741.11
TE-5	169.29	1165.84	520.51	2204.31	177.56
TE-10	1310.00	48490.58	13983.40	43201.62	2207.69
pdh	2652.78	7831.33	6773.78	9663.11	2009.89
di-yuan	210.89	753.78	549.33	1614.56	243.78
dfn-gwin	415.00	1029.33	909.17	787.17	410.00
polska	16.00	31.40	25.00	88.80	14.80
nobel-us	17.75	47.00	28.25	242.50	14.25
Arithmetic Mean	661.60	12411.86	4215.30	13244.02	799.38

Table 3.14: Number of nodes visited for the entire test set.

	cp	cp-i	bc-all	bc-int	bc-5	bc-5-heur	heuristic
CPU time (Arithmetic mean)	1.39	0.81	86.31	9.94	10.46	6.14	1.91
Fraction of total time (Geometric mean)	0.25	1.34	37.59	23.98	29.81	23.41	17.31

Table 3.15: CPU time spent for solving Benders subproblems.

4

Transmission expansion planning with re-design

4.1 Introduction

Long term transmission expansion planning determines, over an horizon of 10 or more years, optimal investments on new transmission lines that make up an economic and reliable electrical network. In its general form, transmission expansion planning is set as a mixed-integer nonlinear stochastic programming problem that minimizes discounted expected costs of investment, subject to constraints depending on uncertain data, such as future growth of electricity demand and of generation.

Historically, transmission expansion planning stems from centralized systems, with both generation and transmission assets belonging to the government. In this setting, transmission planning should ideally be performed jointly with the generation expansion. However, since the resulting optimization problem would be too complex to handle, electrical transmission and energy generation expansion plans are often determined separately, at least for large power systems. Once both expansion plans are available, they can be used as input for some integrated model of generation and transmission, with simplified features. Alternatively, the output of a simplified integrated model can be used as input of the separate expansion planning problems.

The interest of transmission expansion planning also extends to competitive frameworks. The current deregulation trend often results in a mix of market competition in the generation and distribution sectors, with a centralized regulation for transmission. In this context, the regulating entity is in charge, not only of operating the grid while maximizing energy trade opportunities, but also of defining an expansion plan for the transmission network to remain operational in the future. Whether the power system is centralized or liberalized, transmission expansion planning is a valuable tool for helping the decision-maker in adopting the most appropriate strategies for determining the time, the location, and the type of transmission lines to be built.

The transmission expansion planning problem is set over an electrical network, designed in the past by taking into account some critical factors, specific to the power system under consideration. The amount of hydropower is crucial in hydro-dominated power systems like Brazil's, because generation sites are usually far away from the consumption centers. Long transmission lines and, hence, important investments, are needed. Also, due to the pluvial regime, the network needs to accommodate various power flows arising in different hydrological conditions. Another important factor is the demand growth rate along the years, especially for countries with significant growth rates, which need large investments and a large portfolio with reinforcement candidates.

The transmission expansion planning optimization problem includes both physical and budget constraints. Operational and investment constraints are often linear, and vary dynamically along the planning horizon. By contrast, expansion transmission constraints are static and non-convex, generally bilinear. Due to the high complexity and difficulty of the corresponding optimization problem, several simplified models and approximation techniques have been considered; see the review from Latorre et al. (2003). For example, in Tsamasphyrou et al. (2000), the transmission expansion planning problem with security constraints, preventing transmission equipment failure, is set as two-stage stochastic mixed-integer linear program, decomposed by Benders technique and solved by a (multi-cut) cutting-planes algorithm (Birge and Louveaux, 2008) similar to `cp` from Chapter 3. If transmission losses are a concern, they can be treated by a linearization (de la Torre et al., 2008; deOliveira et al., 2005).

Due to the restructuration of the electrical sector that affected many countries in the recent years, uncertainty has lately arisen as an important consideration. This impacts the modeling and significantly increases the size and complexity of the optimization problem. Reported results are mostly for small power systems (6 to 30 buses) Silva et al. (2006); Choi et al. (2007); Maghouli et al. (2009); Buygi et al. (2006); Choi et al. (2005); Fang and Hill (2003); Zhao et al. (2009); Buygi et al. (2004); Lopez et al. (2007); Lu et al. (2005); Reis et al. (2005); de la Torre et al. (2008). When considering larger power systems, the problem size is reduced by some heuristic method, relying on human experts' judgment, as in Tor et al. (2008); Oliveira et al. (2007, 2004b); Buygi et al. (2006).

In general, the transmission expansion planning problem is solved in two variants, considering or not generation redispatch; see Bahiense et al. (2001); de J. Silva Junior (2005); deOliveira et al. (2005). The case without redispatch requires the planned transmission network to operate correctly for a given set of generation values, computed *apriori* for each generation plant. The variant with redispatch considers generation as a variable in the optimization problem: an economic dispatch and the optimal transmission expansion plan are computed together.

In this chapter, published in Moulin et al. (2010), we propose a transmission expansion planning model that, rather than just adding capacity to the existing network, also allows for the network to be re-designed when it is expanded. Our new modeling introduces more flexibility and is general, in the sense that it can be used for different frameworks, with and without redispatch, and independently of the level of simplification or sophistication of the formulation, including with respect to uncertainty treatment.

The new model with re-design relies on the observation that an existing transmission network, designed in the past, may no longer be optimal in the present and it may become even less well adapted in the future. In the transmission expansion planning problem, electrical power flows in the grid according to the linearized second Kirchoff's law, and has the following peculiar property, unique to electrical networks. Namely, in some configurations, disconnecting an existent transmission line (respectively, adding a new line) does not necessarily decrease (respectively, increase) the network capacity. Our numerical testing shows that allowing for the network to be re-designed while expanding it can result in significant savings.

This Chapter is organized as follows. In Section 4.2, we start with a general transmission expansion planning problem, then present our model with re-design, and comment on alternative models proposed by some authors. As mentioned, the transmission expansion planning problem has bilinear constraints that need to be dealt with. Section 4.3 contains a mathematical study comparing different disjunctive proposals that can be found in the literature. Some alternative linearization techniques, improving the relaxed transmission expansion planning problem, are also analyzed. In most of the proposals, bilinear constraints are "linearized" by using the "big-M" reformulation from Disjunctive Programming. The problem of choosing suitable values for the corresponding "big-M" coefficients is addressed in Section 4.4. We first give general minimum values for the models with and without re-design, and then analyze how to exploit the initial network topology to reduce the minimal bounds. Section 4.5 reports on our numerical testing, including a thorough comparison of the various formulations performances on several grids of real size. The final Section 4.6 gives the model with re-design when considering $(N-1)$ security constraints, some preliminary numerical experience, and a discussion on how to handle uncertain demand and generation.

4.2 Models for transmission expansion planning

For convenience, we start by formulating a deterministic transmission expansion planning problem without contingencies; in Section 4.6, we consider how to incorporate $(N-1)$ constraints in the modeling. From the Combinatorial Optimization point of view, the electrical network is an undirected graph (V, E) where vertices $i \in \mathsf{V}$ are called buses and edges $e \in \mathsf{E}$ are called circuits. The set of circuits is partitioned into a subset E^0, of existing circuits, and a disjoint subset of candidate circuits, denoted by E^1. For each circuit $e \in \mathsf{E}$, indices $s(e)$ and $t(e)$ denote, respectively, the head and the tail of the circuit, while ψ_e is the circuit susceptance.

The grid can have parallel circuits, $e_1, e_2 \in \mathsf{E}$, denoted by $e_1 \parallel e_2$, linking the same terminal buses.

4.2.1 Classical transmission expansion planning problem

The transmission network expansion problem usually models the flow x_e on a circuit e by $x_e = \psi_e(\theta_{s(e)} - \theta_{t(e)})$, where potential angles θ_i, $i \in \mathsf{V}$, are decision variables of the problem. Therefore, flow variables x are not needed to formulate the problem, but they are used to simplify the formulation. Each variable g_i states the quantity of electricity generated at node $i \in \mathsf{V}$. Introducing binary design variables y stating which circuits are built, we obtain the following formulation:

$$
(\mathbf{TEP}) \quad
\begin{array}{lllr}
\min & \sum_{e \in \mathsf{E}^1} c_e y_e & & \\
\text{s.t.} & \sum_{e \in \mathsf{E}: t(e)=i} x_e - \sum_{e \in \mathsf{E}: s(e)=i} x_e = d_i - g_i & i \in \mathsf{V} & (4.1) \\
& x_e - \psi_e(\theta_{s(e)} - \theta_{t(e)}) = 0 & e \in \mathsf{E}^0 & (4.2) \\
& x_e - \psi_e y_e(\theta_{s(e)} - \theta_{t(e)}) = 0 & e \in \mathsf{E}^1 & (4.3) \\
& -C \le x \le C & & (4.4) \\
& 0 \le g \le G & & (4.5) \\
& y \text{ binary.} & &
\end{array}
$$

Equations (4.1) are the flow balance constraints at each bus of the network, (4.2) and (4.3) ensure that linearized Kirchoff laws are satisfied for each existing and candidate circuit, respectively, while (4.4) and (4.5) ensure that capacities are not exceeded for each circuit and generating unit. At first glance, problem (**TEP**) could be considered as a Capacitated Network Design problem such as (**ND**), used to model expansion of telecommunication networks (see Chapter 3) and freight transportation networks (Crainic, 2000), among others. However, there is one important difference, that has a crucial impact when solving the transmission expansion planning problem. Specifically, most network design problems, including (**ML**) and (**HOP**), satisfy the following property:

$$
\begin{gathered}
\text{for any given } y \in \{0,1\}^{|\mathsf{E}^1|}, \text{ with components } y_e = \begin{cases} 1 & \text{for } e \in \mathsf{E}' \subset \mathsf{E}^1 \\ 0 & \text{for } e \in \mathsf{E}^1 \backslash \mathsf{E}', \end{cases} \\
\text{if } y \text{ is feasible for a network design problem } (\mathbf{P}), \text{ then any vector } \tilde{y} \in \{0,1\}^{|\mathsf{E}^1|} \\
\text{such that } \tilde{y} \ge y \text{ is also feasible for } (\mathbf{P}).
\end{gathered} \tag{4.6}
$$

Such is not the case for transmission networks. As shown in Figure 4.1, Property (4.6) may not hold for (**TEP**): adding one or more circuits to a functioning network may prevent it from working properly.

This peculiar feature is in sharp contrast with Capacitated Network Design problems. We shall come back to this issue in Section 4.2.2.

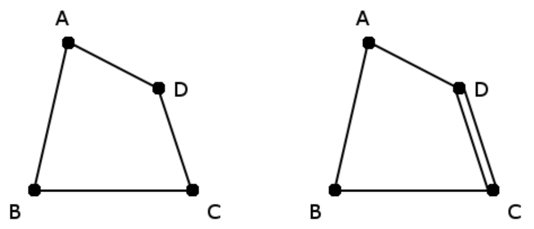


Figure 4.1: $G_A = 100\,MW$, $G_B = G_C = G_D = 0\,MW$, $d_B = d_C = 50\,MW, d_A = d_D = 0\,MW$, $\psi_{AB} = \psi_{BC} = 1\frac{MW}{rad}$ and $\psi_{CD} = \psi_{DA} = 2\frac{MW}{rad}$, and $C_{AB} = C_{BC} = C_{CD} = C_{CA} = 50\,MW$. Left network is feasible for $\theta_A = 0\,rad$, $\theta_B = \theta_C = -50\,rad$ and $\theta_D = -25\,rad$, whereas right network is infeasible.

4.2.2 Allowing for the network to be re-designed

In network design problems, new links are added to a network to make it capable of routing given commodities. Typical examples of commodities are passengers using public transportation, merchandize in a vehicle routing problem, data in a telecommunication network, or electricity in a transmission grid.

As mentioned, the peculiar behavior of power flow makes transmission networks very different from the other examples. In particular, for most network design problems, the routing is either decided by some manager, or fixed by a rule aiming at minimizing some utility (congestion, travel time, travel costs). In such circumstances, the fact of adding a new link to a functioning network can never prevent the network from working properly. At worst, the manager can decide not to use that particular link. By contrast, in transmission power systems, the network manager cannot choose which circuits will be used. Only generation dispatch, indirectly affecting the routing, can be chosen (generation levels are control variables, while voltage angles and flows are state variables). The example in Figure 4.1 shows that, besides being useless, a new link can also make the network inoperational. Similarly, an inoperational network unable to satisfy its load could in some cases start functioning after cutting-off some of its circuits.

The remarks above indicate that, from a modeling point of view, it can be cheaper to allow the network to be re-designed when planning its expansion. The approach is also sensible from a practical point of view. When compared to the high investment required to build new lines, the possibility of cutting some transmission lines, with almost no cost, is worth considering. However, since existing lines can be cut, a model with re-design uses more binary variables and is more difficult from the computational point of view.

The corresponding optimization problem is given by

$$
(\mathbf{TEP_R})\qquad
\begin{aligned}
\min\quad & \sum_{e\in\mathsf{E}^1} c_e y_e && && (4.7)\\
\text{s.t.}\quad & \sum_{e\in\mathsf{E}:t(e)=i} x_e - \sum_{e\in\mathsf{E}:s(e)=i} x_e = d_i - g_i && i\in\mathsf{V} &&\\
& x_e - \psi_e y_e(\theta_{s(e)} - \theta_{t(e)}) = 0 && e\in\mathsf{E} && (4.8)\\
& -C \le x \le C &&&&\\
& 0 \le g \le G &&&&\\
& y \text{ binary.} &&&&
\end{aligned}
$$

Problem ($\mathbf{TEP_R}$) has one variable y_e for each circuit $e \in \mathsf{E}$ while (**TEP**) has a variable y_e only for $e \in \mathsf{E}^1$. Therefore, the bilinear constraints (4.8) in ($\mathbf{TEP_R}$) are set for all circuits, not only for the new ones as in (4.3). Both problems have the same objective function (4.7): only investment cost in building new lines is considered, because the cost of cutting an existing line is negligible. Note, in addition, that the classical model (**TEP**) can be derived from ($\mathbf{TEP_R}$), by adding the constraints $y_e = 1$ for $e \in \mathsf{E}^0$ to the re-design problem. This unified approach will be useful in the sequel, when devising solution methods.

In addition to having more binary variables, model ($\mathbf{TEP_R}$) is harder to solve than (**TEP**) because some of the binary variables have null objective cost. As a result, when using an enumeration method, the fathoming of many nodes in the branch-and-bound tree can be significantly delayed. For the same reason, meta-heuristics providing very good feasible solutions for (**TEP**), such as the GRASP described by Binato et al. (2001), are no longer applicable to ($\mathbf{TEP_R}$), because they are based on selecting circuits by the corresponding investement cost. Finally, as shown in Section 4.4, the linear relaxation polyhedron for ($\mathbf{TEP_R}$) is larger than the one of (**TEP**). As a result, bounds for ($\mathbf{TEP_R}$) may be less tight than for (**TEP**).

Despite the apparently negative comments above, it is important to keep in mind that, depending on the particular problem, allowing for re-design may have a significant economic impact. Our numerical results on real-life transmission networks show that the model with re-design gives important savings for some configurations.

4.2.3 Simplified related models

Both (**TEP**) and ($\mathbf{TEP_R}$) can be further complicated by the introduction of $(N-1)$ security constraints. These constraints state that if, for some contingency, any circuit happens to fail (alone), the network must stay functional. We will come back to this issue in Section 4.6.

In view of the difficulty of the transmission expansion planning problem, even without contingencies, several authors introduced simplified models that we review next. However, in all of the models below, simplification comes at the stake of ending up with a network for which (4.6) holds. Since this property is not satisfied by a transmission network, for some applications the (simplified) optimal plan computed with such models may need to be modified when the network is actually expanded.

In Bienstock and Mattia (2007), the model is set to find a minimal cost capacity increase that ensures the network survival to different failures. The network is represented by a graph without parallel edges, and each edge e has an initial capacity, denoted by u_e. Parallel circuits are summed up into a single edge with the corresponding total capacity. In the absence of parallel edges, bilinear constraints can be avoided by replacing, for all $e \in \mathsf{E}$, constraints (4.8) and (4.4) by

$$x_e - \psi_e(\theta_{s(e)} - \theta_{t(e)}) = 0 \qquad \text{and} \qquad |x_e| \leq u_e + C_e y_e \,,$$

respectively. Failures are considered in two different variants, depending if they occur simultaneously or in cascade. The first variant is solved by an efficient Benders decomposition scheme. The solution method for the second variant makes use of strong valid inequalities in a cutting planes framework. For both variants, the elimination of parallel circuits allows the authors to solve much bigger instances than the ones handled in our numerical results.

Another simplified model goes back to Garver's transportation model (Garver, 1970), where (4.3) is replaced by a flow constraint of the form $|x_e| \leq y_e C_e$ for all $e \in \mathsf{E}^1$. The resulting mixed-integer linear programming problem is easy to solve by modern solvers, because it is closely related to the so-called single-commodity multi-facility capacitated network design problem. Although unrealistic, the transportation model can provide a better lower bound for (**TEP**) and ($\mathbf{TEP_R}$) than the optimal value of the linear relaxation, see Table 4.5 in Section 4.5. Hence, it can be efficiently used in a branch-and-bound process to eliminate portions of the exploration tree.

The third model in our review was proposed by Singh et al. (2008) for electricity distribution. Due to the local span of distribution networks, there is one generating unit (only one generation bus) and the network must be a tree (each pair of buses is connected by a single path). In this setting, the model is no longer a simplification, because the actual network satisfies (4.6).

The tree requirement introduces many combinatorial affine constraints. In counterpart, we show below that a tree network makes the (bilinear) second Kirchoff's law redundant, simplifying substantially the optimization problem (voltage angles disappear from the formulation) when C and ψ satisfy

$$e_1 \parallel e_2 \Rightarrow C_{e_1}/\psi_{e_1} = C_{e_2}/\psi_{e_2}, \qquad e_1, e_2 \in \mathsf{E}. \tag{4.9}$$

Proposition 4.1 (Consequence of tree shape)**.** *Let Garver's transportation model* *be given by*

$$
\textbf{(Transportation)} \qquad
\begin{aligned}
\min \quad & \sum_{e \in \mathsf{E}^1} c_e y_e \\
s.t. \quad & \sum_{e \in \mathsf{E}: t(e)=i} x_e - \sum_{e \in \mathsf{E}: s(e)=i} x_e = d_i - g_i && i \in \mathsf{V} \\
& -C_e y_e \le x_e \le C_e y_e && e \in \mathsf{E}^1 \\
& -C \le x \le C \\
& 0 \le g \le G \\
& y \textit{ binary.}
\end{aligned}
\tag{4.10}
$$

If C and ψ satisfy (4.9) *and y designs a tree, then any vector (y, x, g) is feasible for the transportation problem above if and only if there exists a vector (y, x', g, θ') feasible for the transmission expansion planning (***TEP***).*

Proof. The necessary condition is straightforward, because the feasible set of the transmission expansion planning (**TEP**) is contained in the feasible set of the transportation model. To prove the reverse inclusion, given a tree y and a vector (y, x, g) feasible for the transportation model, we define a vector (y, x', g', θ') that is feasible for (**TEP**), as follows. First, we keep the same generation variables, $g'_i = g_i$ for each $i \in \mathsf{V}$. Then, we consider any circuit $e_1 \in \mathsf{E}$ with endpoints i and j. The total flow between i and j is bounded by the total capacity of the circuits connecting i and j, so that their ratio X_{ij} is smaller than one:

$$X_{ij} \equiv \frac{\sum_{e_2 \in \mathsf{E}: e_2 \| e_1} x_{e_2}}{\sum_{e_2 \in \mathsf{E}^1 : e_2 \| e_1} y_{e_2} C_{e_2} + \sum_{e_2 \in \mathsf{E}^0 : e_2 \| e_1} C_{e_2}} \le 1.$$

Then, $x'_{e_1} = C_{e_1} X_{ij} \le C_{e_1}$ for $e_1 \in \mathsf{E}^0$ and $x'_{e_1} = y_{e_1} C_{e_1} X_{ij} \le C_{e_1}$ for $e_1 \in \mathsf{E}^1$ so that x' satisfies constraints (4.4). The constraint (4.1) for any $i \in \mathsf{V}$ is also satisfied, because g_i is equal to g'_i and the total flow from i to any $j \in \mathsf{V}$ is unchanged: for any $e_1 \in \mathsf{E}$ such that $s(e_1) = i$ and $t(e_1) = j$, the total flow between i and j is given by

$$\sum_{e_2 \in \mathsf{E}: e_2 \| e_1} x'_{e_2} = X_{b_1 b_2} \left(\sum_{e_2 \in \mathsf{E}^1 : e_2 \| e_1} y_{e_2} C_{e_2} + \sum_{e_2 \in \mathsf{E}^0 : e_2 \| e_1} C_{e_2} \right) = \sum_{e_2 \in \mathsf{E}: e_2 \| e_1} x_{e_2}.$$

The new flow vector x' allows us to set up feasible voltage angles θ' satisfying (4.2) and (4.3), as follows. First, we choose any bus $ref \in \mathsf{V}$ and set $\theta'_{ref} = 0$. Then, we select any built circuit e_1 ($e_1 \in \mathsf{E}^1$ and $y_{e_1} = 1$, or $e_1 \in \mathsf{E}^0$) with $s(e_1) = ref$ and set $\theta'_{t(e_1)} = \theta'_{s(e_1)} - x'_{e_1}/\psi_{e_1} = 0 - x'_{e_1}/\psi_{e_1} = -X_{s(e_1)t(e_1)} C_{e_1}/\psi_{e_1}$. Assumption (4.9) ensures that choosing $e_2 \| e_1$, instead of e_1, induces the same angles difference. Next, we select a built circuit $e_2 \nparallel e_1$ with $s(e_2) \in \{ref, t(e_1)\}$ to set up $\theta'_{j(e_2)}$ in the same way. We repeat this procedure until all voltage angles are set, the tree shape ensuring that each of them shall be set only once. $\square$

The fact that transmission networks with a tree shape do not require potential angle variables allows us to reduce the Steiner-tree problem to (**TEP**).

Corollary 4.2 (Complexity of transmission expansion planning). *Problem (***TEP***) is $\mathcal{NP}$-hard.*

Proof. We show how to write a Steiner-tree graph problem in the form (**TEP**), by suitably choosing the parameters therein. Given an undirected weighted graph defined by vertices in a set V and edges in a set E, a set of terminal vertices $\mathsf{T} \subseteq \mathsf{E}$, with $|\mathsf{T}| \geq 3$, and edge weights $c_e \geq 0$ for all $e \in \mathsf{E}$, the *Steiner Problem in Graphs* consists in finding a connected subgraph S (called the *Steiner Tree*) that includes all terminal vertices at minimum edge cost, i.e., $\min \sum_{e \in \mathsf{E}} c_e$. This problem is known to be $\mathcal{NP}$-hard, especially for grid graphs (Karp, 1972; Ahuja et al., 1993). Likewise for the single-commodity flow integer formulation of the Steiner problem (Wong, 1984). This formulation expresses the original (undirected weighted graph) problem as a directed weighted graph problem by choosing a "source" terminal vertex t_s offering commodities to the remaining terminal vertices. To see how this last formulation can be cast in the form of problem (**TEP**), we consider the the same graph (V, E) and let $\mathsf{E} = \mathsf{E}^1$ and $\mathsf{E}^0 = \phi$. Then, if t denotes the cardinality of the set of terminal vertices $\mathsf{T} \subset \mathsf{V}$, for an arbitrary source $t_s \in \mathsf{T}$, we take

$$G_i = \begin{cases} t-1 & i = t_s \\ 0 & i \text{ in } \mathsf{V}\backslash\{t_s\} \end{cases}, \qquad d_i = \begin{cases} 0 & i = \text{ in } \mathsf{V}\backslash\mathsf{T} \cup \{t_s\} \\ 1 & i \text{ in } \mathsf{T}\backslash\{t_s\} \end{cases},$$

and, for all $e \in \mathsf{E}$, $C_e \geq t - 1$ and $\psi_e = 1$. Because the capacities are large enough and designs c costs are positive, an optimal solution to (**TEP**) has a tree shape. Therefore, potential angles may be neglected and the solution is nothing but a minimum cost Steiner tree connecting vertices in T. □

4.3 Linearizing the problem

We now address the problem of defining tight and convex relaxations for the mixed-integer bilinear programming problem ($\mathbf{TEP_R}$). Since (**TEP**) can be formulated as ($\mathbf{TEP_R}$) plus constraints $y_e = 1$ for $e \in \mathsf{E}^0$, the formulations below can be used for both models.

The main difficulty of ($\mathbf{TEP_R}$) arises from its bilinear constraints (4.8), defining the function

$$F(y_e, \theta_{s(e)}, \theta_{t(e)}) := \psi_e y_e (\theta_{s(e)} - \theta_{t(e)}) \,.$$

This is a bilinear function, neither convex nor concave (its Hessian eigenvalues are constant, equal to 0 and to $\pm\sqrt{2}\psi_e$). Moreover, there is no quadratic convexification for $F(y_e, \theta_{s(e)}, \theta_{t(e)})$, because the function $F(y_e, \theta_{s(e)}, \theta_{t(e)}) + \lambda(y_e^2 - y_e)$, with Hessian eigenvalues equal to 0 and to $\lambda \pm \sqrt{\lambda^2 + 2\psi_e^2}$, remains neither convex nor concave, regardless the value of the scalar λ. For this reason, efficient convex mixed-integer nonlinear programming tools, like the method from Quesada and Grossman (1992) and its modern implementation FilMint from Abhishek et al., cannot be used in our problem.

Instead, bilinear constraints are "linearized" by using the "big-M"-reformulations for disjunctive programming (Raman and Grossmann, 1994). Before detailing how to suitably choose such coefficients, we compare two disjunctive approaches that have been used in the literature and give an alternative, third, formulation using "big-M" constraints. To each one of the three formulations corresponds a specific rewriting of bilinear constraints, that yields a different optimization problem, depending if the model of interest is (**TEP**) or ($\mathbf{TEP_R}$).

4.3.1 Standard Disjunctive Formulation

Pereira and Granville (1985), and Villanasa (1984), among others, replace (4.8) by a constraint of the form

$$-M_e(1-y_e) \le x_e - \psi_e(\theta_{s(e)} - \theta_{t(e)}) \le M_e(1-y_e) \quad \text{for all } e \in \mathsf{E}\,, \tag{4.11}$$

for some fixed coefficients $M_e > 0$. Flow bounds are written in the form

$$|x_e| \le y_e C_e \qquad \text{for all } e \in \mathsf{E}\,. \tag{4.12}$$

The advantage of this formulation is that its number of variables and constraints grows linearly with the size of the problem. Yet, the formulation is very hard to solve because of the "big-M" coefficients in constraints (4.11).

4.3.2 Improved Disjunctive Formulation

A new disjunctive formulation, hopefully tighter than the standard one, and requiring additional continuous variables, was considered by Bahiense et al. (2001). Each flow is rewritten by using two positive flow variables, as follows:

$$x_e = x_e^+ - x_e^- \qquad \text{for } x_e^+, x_e^- \ge 0 \text{ and } e \in \mathsf{E}\,. \tag{4.13}$$

Likewise for each voltage angles difference:

$$\Delta\theta_e^+ - \Delta\theta_e^- = \theta_{s(e)} - \theta_{t(e)} \qquad \text{for } \Delta\theta_e^+, \Delta\theta_e^- \ge 0 \text{ and } e \in \mathsf{E}\,. \tag{4.14}$$

Using the additional variables in (4.11) yields the following constraints

$$\begin{array}{l} -M_e(1-y_e) \le x_e^+ - \psi_e \Delta\theta_e^+ \le 0 \\ -M_e(1-y_e) \le x_e^- - \psi_e \Delta\theta_e^- \le 0 \end{array} \quad \text{for all } e \in \mathsf{E}\,. \tag{4.15}$$

With the new variables, flow bounds take the form

$$x_e^+ \le y_e C_e \qquad \text{and} \qquad x_e^- \le y_e C_e \quad \text{for all } e \in \mathsf{E}\,. \tag{4.16}$$

The relation expressing a variable as the difference of its positive and negative parts is a bijection. For this reason, (4.13) and (4.16) are equivalent to (4.12). Since, rather than using the voltage angles, the bijection is used for the voltage angles differences in (4.14), the feasible set defined by (4.15) differs from the one defined by (4.11), as shown next.

Comparing linear relaxations

An important matter when relaxing mixed-integer constraints refers to how close the new feasible set is to the convex hull of the original feasible set, see (Nemhauser and Wolsey, 1999). A formulation for which the relation is tight is said to be *stronger* than one with a bigger set. To compare the strength of the disjunctive formulations above, we consider their *linear relaxation* polyhedrons, obtained when replacing the $\{0,1\}$ set by the interval $[0,1]$.

Accordingly, we define the polyhedrons

$$\mathcal{P} = \text{conv}\left(\left\{(y,x,g,\theta) \text{ satisfies (4.1), (4.8), (4.4), (4.5) and } y \in \{0,1\}^{|\mathsf{E}|}\right\}\right) ,$$

corresponding to the convex hull of feasible vectors for model $(\mathbf{TEP_R})$;

$$\mathcal{P}_{4.3.1} := \left\{(y,x,g,\theta) \text{ satisfies (4.1), (4.11), (4.12), (4.5) and } y \in [0,1]^{|\mathsf{E}|}\right\} ,$$

corresponding to the linear relaxation of the standard disjunctive formulation of model $(\mathbf{TEP_R})$; and

$$\mathcal{P}_{4.3.2} := \left\{(y,x,g,\theta) \text{ satisfies } \begin{bmatrix} \text{(4.1), (4.13), (4.14),} \\ \text{(4.15), (4.16), (4.5)} \end{bmatrix} \text{ and } y \in [0,1]^{|\mathsf{E}|}\right\} ,$$

corresponding to the linear relaxation of the improved disjunctive formulation of model $(\mathbf{TEP_R})$.

We first note that the improved disjunctive formulation is tighter than the standard one. More precisely, in (4.15), substracting the second equation from the first one, and using (4.13) and (4.14), implies satisfaction of (4.11). Therefore,

$$\mathcal{P}_{4.3.2} \subseteq \mathcal{P}_{4.3.1} \cdot \tag{4.17}$$

The following example shows that the inclusion may be strict.

Example 4.3 (Strict inclusion)**.** Consider a network formed by three buses $A\,,B$, and C, with no initial circuits and such that at most one circuit connecting each pair of buses can be built. Suppose, in addition, that the parameters have the values $G_A = 100\,MW$, $G_B = G_C = 0$, $d_A = d_C = 0$, $d_B = 100\,MW$, $C_{AB} = C_{BC} = C_{CA} = 400\,MW$, $\psi_{AB} = \psi_{CA} = 1\frac{MW}{rad}$ and $\psi_{BC} = 0.5\frac{MW}{rad}$ and $c_{AB} = c_{BC} = c_{CA} = 10$. The optimal value to the transmission expansion planning optimization problem $(\mathbf{TEP_R})$ is 10, obtained by constructing only circuit AB: $y_{AB} = 1$, the remaining optimal binary variables being null. The corresponding voltage angles at the optimum are $\theta_A = 0\,rad$ and $\theta_B = -100\,rad$. We show next how to construct a cheaper fractional solution (y,x,g,θ) in $\mathcal{P}_{4.3.1}$ that does not belong to $\mathcal{P}_{4.3.2}$.

In Section 4.4 we give the smallest values for the "big-M" coefficients to ensure a tight relaxation. In particular, by Proposition 4.6 therein, the minimal value for M_{BC} is $400\,MW$. Consider the fractional vector $y_{AB} = y_{BC} = y_{CA} = 0.25$, with angles $\theta_A = 0$, $\theta_B = \theta_C = -50\,rad$ and flows $x_{AC} = x_{CB} = x_{AB} = 50\,MW$. The corresponding objective function value is 7.5, smaller than the optimal cost of the mixed 0-1 problem.

For the point under consideration, the potential differences $\psi_{AB}(\theta_A - \theta_B) = \psi_{AC}(\theta_A - \theta_C) = 50\,MW$ are enough to induce the required flows, whereas $\psi_{CB}(\theta_C - \theta_B) = 0\,MW$ should not induce any flow. However, since y is fractional, the "big-M" constraints may allow this flow to be routed on the network. Namely, constraint (4.11) for circuit CB is

$$-300 \leq x_{CB} \leq 300\,,$$

while constraint (4.15) for circuit CB is

$$x_{CB} = 0. \tag{4.18}$$

Thus, the flows $x_{AC} = x_{CB} = x_{AB} = 50\,MW$ give a feasible point in $\mathcal{P}_{4.3.1}$. By contrast, constraints (4.18) will cut-off the point from $\mathcal{P}_{4.3.2}$. $\square$

For a linear relaxation to be useful for the optimization problem, its "shadow" projection on the y-variables (Nemhauser and Wolsey, 1999) needs to be tight with respect to the original problem. This means that in the relaxed polyhedrons only the y-components of feasible vectors (y, x, g, θ) matter.

In this sense, although the inclusion (4.17) ensures a similar relation for the shadow projections, we are in no position to say if the inclusion is strict for the y-variables only. In particular, we now show that for the counter-example above, it is possible to define flows and angles $\tilde{x}, \tilde{\theta}$ satisfying (4.15) for the fractional values $y_{AB} = y_{BC} = y_{CA} = 0.25$.

Example 4.4 (No longer strict inclusion)**.** Consider the network in Example 4.3 and the same fractional vector $y_{AB} = y_{BC} = y_{CA} = 0.25$. Set angles to $\tilde{\theta}_A = \tilde{\theta}_C = 0\,rad$ and $\tilde{\theta}_B = -100\,rad$, and flows to $\tilde{x}_{AB} = 100\,MW$, $\tilde{x}_{AC} = \tilde{x}_{CB} = 0\,MW$. Such flows $\tilde{x}_{AB}$ and $\tilde{x}_{AC}$ are correctly induced by the potential differences, as long as the flow $\tilde{x}_{CB}$ is equal to $50\,MW$. However, recalling that $M_{BC}(1 - y_{BC}) = 300\,MW$, constraint (4.15) for circuit CB is

$$-250 \leq x_{CB} \leq 50, \tag{4.19}$$

so that $\tilde{x}_{CB} = 0\,MW$ is feasible for (4.19) and $(y, \tilde{x}, \tilde{\theta}, g) \in \mathcal{P}_{4.3.2}$. $\square$

In summary, from relation (4.17), the linear relaxation of the improved disjunctive formulation is not worse than the one of the standard disjunctive formulation. But it is not known if, when considering only the y-components, the inclusion remains strict (unfortunately, no example is given by Bahiense et al. (2001)). In our computational experience in Section 4.5, both disjunctive formulations gave identical results, for all the cases in Table 4.1.

4.3.3 Breaking Symmetry

In Combinatorial Optimization, it is well known that feasible sets exhibiting symmetry often slow down significantly any branch-and-bound algorithm, due to (useless) exploration of many symmetric nodes. In a transmission network, parallel circuits do induce such a symmetry, making both disjunctive formulations in Sections 4.3.1 and 4.3.2 difficult to solve.

Basically, parallel circuits yield feasible points that are indistinguishable by the objective function. Indeed, from a feasible vector involving parallel circuits $e_1 \parallel e_2$, another feasible vector with the same cost can be obtained, simply by swapping indices corresponding to e_1 and e_2.

In order to address this important issue, in what follows we assume the condition below.

Assumption 4.5. *Any pair of parallel circuits $e_1, e_2 \in \mathsf{E}$ has the same capacity, susceptance and cost:*

$$\forall e_1 \parallel e_2 \qquad C_{e_1} = C_{e_2}\,, \quad \psi_{e_1} = \psi_{e_2}\,, \ \textit{and}\ c_{e_1} = c_{e_2}\,.$$

All the case studies considered in our numerical experience, and given in Table 4.1 below, satisfy Assumption 4.5.

The interest of Assumption 4.5 is that it allows us to define new circuit sets, by gathering parallel circuits into a single, "fat", edge. We denote such new sets by F^0 and F^1, corresponding to E^0 and E^1, respectively, with $\mathsf{F} = \mathsf{F}^0 \cup \mathsf{F}^1$ associated to the full set E. This re-ordering does not prevent the network from having parallel circuits: to each (undirected) "fat" edge $ij \in \mathsf{F}$ we associate an upper bound N_{ij} for the number of circuits that can be built. We also denote by n_{ij} the initial number of circuits linking i and j. With this notation, instead of using a single index e for a circuit and terminal points $s(e)$ and $t(e)$, each circuit is now determined by a pair (ij, ℓ), referring to the circuit's endpoints $i, j \in \mathsf{V}$ and the circuit position $\ell \in \mathsf{L}_{ij} := \{1, \dots, n_{ij} + N_{ij}\}$; see Figure 4.2.

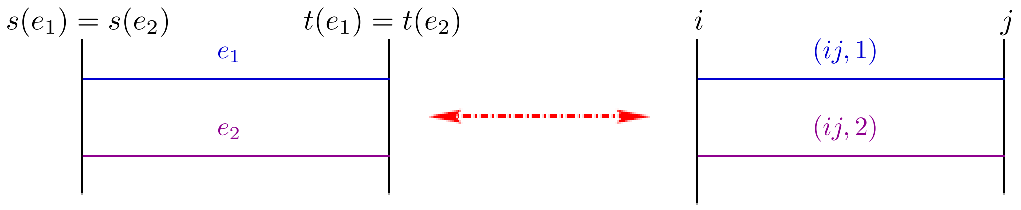


Figure 4.2: Renaming parallel circuits as part of a single, "fat", edge.

Variables y_e and x_e are renamed accordingly to y_{ij}^ℓ and x_{ij}^ℓ, and similarly for the investment costs. We show in Section 4.4 that the actual value used for M_{ij}^ℓ is independent of ℓ, so that constraints (4.11) and (4.15) are rewritten

$$-M_{ij}(1 - y_{ij}^\ell) \leq x_{ij}^\ell - \psi_{ij}(\theta_i - \theta_j) \leq M_{ij}(1 - y_{ij}^\ell) \quad \text{for all } ij \in \mathsf{F}, \ell \in \mathsf{L}_{ij}\,, \qquad (4.20)$$

and

$$\begin{array}{l} -M_{ij}(1-y_{ij}^{\ell}) \le x_{ij}^{\ell+} - \psi_{ij}\Delta\theta_{ij}^{+} \le 0 \\ -M_{ij}(1-y_{ij}^{\ell}) \le x_{ij}^{\ell-} - \psi_{ij}\Delta\theta_{ij}^{-} \le 0 \end{array} \quad \text{for all } ij \in \mathsf{F}, \ell \in \mathsf{L}_{ij}\,, \tag{4.21}$$

respectively.

Symmetry in the disjunctive formulations can be broken in two different ways:

By ordering parallel candidate circuits: a second circuit can be built only if the first one has been built, and so on:

$$y_{ij}^{\ell+1} \le y_{ij}^{\ell} \qquad ij \in \mathsf{F}, \ell, \ell+1 \in \mathsf{L}_{ij}\,. \tag{4.22}$$

These constraints seem to be what Oliveira et al. (2004a) called "Logical precedence" constraints.

By introducing lexicographical costs: a drawback of the ordering above is the resulting increase in the number of constraints. Instead, parallel circuits can be made distinguishable (and ordered) by assigning to each one of them a different cost, depending on some positive constant ϵ, possibly small:

$$\begin{array}{ll} c_{ij}^{\ell} = (\ell-1)\epsilon & ij \in \mathsf{F}\,, 1 \le \ell \le n_{ij} \\ c_{ij}^{\ell} = c_{ij} + (\ell-1)\epsilon & ij \in \mathsf{F}\,, n_{ij}+1 \le \ell \in \mathsf{L}_{ij}\,. \end{array} \tag{4.23}$$

In our numerical tests, the improved disjunctive formulation from Section 4.3.2 did not give competitive results. For this reason, we applied (4.22) and (4.23) only to the standard disjunctive formulation from Section 4.3.1. The lexicographical ordering (4.23) turned out to be rather poor, at least in our case studies. For instance, the transmission expansion planning for the network "Brazil South", modeled by (**TEP**) and using the standard disjunctive formulation, took 5 seconds to be solved until optimality. When introducing (4.23), solution times climbed up to more than 300 seconds.

We mention that CPLEX 11 has an automatic symmetry breaking procedure which can sensibly affect solution times. When this procedure is deactivated, by setting *IntParam.Symmetry* = 0, the standard disjunctive formulation is much slower than when using the approaches (4.22) or (4.23). However, when setting *IntParam.Symmetry* = -1, the impact of (4.22) becomes much less expressive.

4.3.4 Alternative Disjunctive Formulation

We also considered a third disjunctive formulation, grouping together parallel circuits:

$$
\begin{array}{lll}
\min & \displaystyle\sum_{ij\in \mathsf{F}^1} c_{ij} \sum_{\mathsf{L}_{ij}\ni \ell \geq n_{ij}+1} (\ell - n_{ij})\, y_{ij}^{\ell} & \\
\text{s.t.} & \displaystyle\sum_{ji\in \mathsf{F}} x_{ji} - \sum_{ij\in \mathsf{F}} x_{ij} = d_i - g_i & i\in \mathsf{V} \\
& -M_{ij}^{\ell}(1-y_{ij}^{\ell}) \leq \dfrac{x_{ij}}{\ell} - \psi_{ij}(\theta_i - \theta_j) \leq M_{ij}^{\ell}(1-y_{ij}^{\ell}) & ij\in \mathsf{F}, \ell\in \mathsf{L}_{ij} \quad (4.24)\\
& \displaystyle\sum_{\ell\in \mathsf{L}_{ij}} y_{ij}^{\ell} \leq 1 & ij\in \mathsf{F} \quad (4.25)\\
& -C_{ij}\displaystyle\sum_{\ell\in \mathsf{L}_{ij}} \ell\, y_{ij}^{\ell} \leq x_{ij} \leq C_{ij}\sum_{\ell\in \mathsf{L}_{ij}} \ell\, y_{ij}^{\ell} & ij\in \mathsf{F} \quad (4.26)\\
& 0 \leq g \leq G & \\
& y \text{ binary.} &
\end{array}
$$

Although this formulation significantly reduces the number of flow variables, such potential advantage was not reflected in our computational results; see Section 4.5. In an effort to improve performance, we also tried CPLEX functionality of using type SOS 1 constraints instead of (4.25) above. But this option was not effective, probably due to the important increase in the number of nodes to be explored. Setting different values to *IntParam.Symmetry* did not bring much benefit either.

4.4 Choosing suitable "big-M" coefficients

The efficient solution of the linearized disjunctive formulations depends strongly on how the "big-M" coefficients are set. Bigger coefficients give less tight polyhedrons, and worse optimal relaxation values. It is then worthwhile to compute minimal values, $\underline{M}_{ij}$, such that constraints (4.20), (4.21) and (4.24) above are valid for $\mathcal{P}$, for any given value of G and d. Recall that an inequality is said to be valid for a polyhedron if the polyhedron is contained in the half space delimited by the inequality.

We first give general minimum values for the models with and without re-design, and then analyze how to exploit the initial network topology (F^0) to reduce the minimal bounds.

In our analysis, paths are always assumed without cycles (they cannot contain twice the same node).

We start with model ($\mathbf{TEP_R}$), allowing re-design, noting that, for any vector $y \in \{0,1\}^{|\mathsf{F}|}$, when $G = d = 0$, the point $(y, x = 0, \theta = 0, g = 0)$ trivially belongs to $\mathcal{P}$. For this reason, the bound for the "big-M" coefficients should be found for any binary vector x. We now show that the computation of such bound involves solving a longest path problem (Hardgrave and Nemhauser, 1962).

Proposition 4.6. *Suppose Assumption 4.5 holds and let ij be given. Consider constraints* (4.20) *and* (4.21) *from Section 4.3.3. Then the minimal admissible value for M_{ij} such that these constraints are valid for $\mathcal{P}$, for any $G, d \geq 0$, is given by $\underline{M}_{ij}^{(\mathbf{TEP_R})} = \psi_{ij} LP_{i-j}$, where LP_{i-j} denotes the length of the longest path between the buses i and j, computed with costs*

$$\tilde{c}_{b_1 b_2} = \frac{C_{b_1 b_2}}{\psi_{b_1 b_2}}. \tag{4.27}$$

Proof. For given $ij \in \mathsf{F}$ and $\ell \in \mathsf{L}_{ij}$, when $y_{ij}^{\ell} = 1$, constraints (4.20) and (4.21) imply that $x_{ij}^{\ell} = \psi_{ij}(\theta_i - \theta_j)$, regardless the value of M_{ij}. Therefore, we only need to consider $y_{ij}^{\ell} = 0$. The flow bounds (4.12) and (4.16) force $x_{ij}^{\ell} = 0$. Since the corresponding constraints (4.20) and (4.21) state that

$$M_{ij} \geq \psi_{ij} |\theta_i - \theta_j| \,,$$

we just need to find the largest value of $|\theta_i - \theta_j|$, among all possible network configurations having $x_{ij}^{\ell} = 0$.

To this aim, take $n \neq \ell$ in L_{ij} and set $y_{ij}^{n} = 1$. The flow on (ij, n) is at most C_{ij}, so (4.20) and (4.21) written for the circuit (ij, n) imply that $\psi_{ij}|\theta_i - \theta_j| \leq C_{ij}$. Thus, M_{ij} must be at least greater than C_{ij}.

Now, set $y_{ij}^{n} = 0$ for all $n \in \mathsf{L}_{ij}$ such that for some path p between i and j and any link $b_1 b_2$ in the path it holds that $\sum_n y_{b_1 b_2}^{n} \geq 1$. Then, regardless the other values of y, the difference $|\theta_i - \theta_j|$ cannot be greater than:

$$\sum_{b_1 b_2 \in p} |\theta_{b_1} - \theta_{b_2}| \leq \sum_{b_1 b_2 \in p} \frac{C_{b_1 b_2}}{\psi_{b_1 b_2}}.$$

Since we do not know in advance which path will satisfy the relation $\sum_n y_{b_1 b_2}^{n} \geq 1$, we must take the maximum over all paths between i and j. Therefore, $|\theta_i - \theta_j| \leq LP_{i-j}$, with costs given by (4.27). We are left to show that LP_{i-j} is a minimal value to obtain $\underline{M}_{ij}^{(\mathbf{TEP_R})} = \psi_{ij} LP_{i-j}$.

To see that LP_{i-j} is a minimal value, it is enough to show that for any path p between i and j,

$$\theta_i - \theta_j = \sum_{b_1 b_2 \in p} \frac{C_{b_1 b_2}}{\psi_{b_1 b_2}} \tag{4.28}$$

for at least one generation vector G and one demand vector d. Define G and d as follows: for each $b_1 b_2 \in p$, $G_{b_1} = d_{b_2} = C_{b_1 b_2}$, and $G_r = d_r = 0$ otherwise. Thus, $\theta_{b_1} = \theta_{b_2} + \frac{C_{b_1 b_2}}{\psi_{b_1 b_2}}$ for each $b_1 b_2 \in p$, yielding (4.28). □

Note that if the only path from i to j is given by a single candidate circuit from i to j, i.e., by $(ij, 1)$, then there are no constraints on M_{ij} and we can just take $\underline{M}_{ij}^{(\mathbf{TEP_R})} = 0$.

The computation of the minimal value for the third disjunctive formulation can be done in a similar manner.

Corollary 4.7. *Suppose Assumption 4.5 holds and let ij be given. Consider constraint (4.24) from Section 4.3.4. Then the corresponding minimal admissible value for M_{ij}^{ℓ}, for any $G, d \geq 0$, is given by $\underline{M}_{ij}^{(\mathbf{TEP_R}),\ell} = \psi_{ij} LP_{i-j}$, the length of the longest path between the buses i and j computed with the costs (4.27) for $b_1b_2 \neq ij$, and the following cost for ij*

$$\tilde{c}_{ij}^{\ell} = \begin{cases} \frac{n_{ij}+N_{ij}-\ell}{\ell} \frac{C_{ij}}{\psi_{ij}} & \ell \in \mathsf{L}_{ij}\,, \ell \neq n_{ij}+N_{ij} \\ \frac{n_{ij}+N_{ij}-1}{n_{ij}+N_{ij}} \frac{C_{ij}}{\psi_{ij}} & \ell = n_{ij}+N_{ij}\,. \end{cases} \tag{4.29}$$

Proof. When $y_{ij}^{\ell} = 1$, (4.24) forces $x_{ij} = \ell\psi_{ij}(\theta_i - \theta_j)$, for any value of M_{ij}^{ℓ}. Consider then that $y_{ij}^{\ell} = 0$ for all $n \in \mathsf{L}_{ij}$. By constraint (4.26) in Section 4.3.4, $x_{ij} = 0$ and, like in Proposition 4.6, constraint (4.24) becomes $M_{ij} \geq \psi_{ij}|\theta_i - \theta_j|$. As a result, M_{ij}^{ℓ} must be at least greater than the length of the longest path between i and j, in the graph $\mathsf{F}\backslash\{ij\}$.

Otherwise, let $y_{ij}^{n} = 1$ for some $n \in \mathsf{L}_{ij}$, with $n \neq \ell$. Then, $x_{ij} = n\psi_{ij}(\theta_i - \theta_j)$, so that (4.24) written for (ij, ℓ) is

$$M_{ij}^{\ell} \geq \left|\psi_{ij}\left(\frac{n}{\ell} - 1\right)(\theta_i - \theta_j)\right| = \psi_{ij}\frac{|n-\ell|}{\ell}|\theta_i - \theta_j|\,. \tag{4.30}$$

This value cannot be greater than $\frac{|n-\ell|}{\ell}C_{ij}$. If $\ell < n_{ij} + N_{ij}$, the right-hand-side of (4.30) is maximized when $n = n_{ij} + N_{ij}$. If $\ell = n_{ij} + N_{ij}$, the right-hand-side of (4.30) is maximized for $n = 1$. □

Note that, even though the shortest path problem is polynomial and can be solved efficiently by -for instance- Dijkstra's algorithm, the situation is quite different for the longest path. For a graph containing cycles, for example, the problem can be $\mathcal{NP}$-hard. Otherwise, if we could compute in polynomial time the longest path between two adjacent nodes i and j, not passing trough ij, with all arc lengths set to 1, we would also be able to find out in polynomial time whether the graph has a Hamiltonian cycle (this is an $\mathcal{NP}$-complete problem (Ahuja et al., 1993)).

The longest path value LP_{i-j} is often so big that Kirchoff's second law usually fails to hold for the relaxed optimal solution when design variables y are fractional. We now show that the bound can be substantially improved for model (**TEP**), without re-design (and, hence, with less 0-1 variables than ($\mathbf{TEP_R}$)).

Because we extend an existing transmission network, we assume the condition below.

Assumption 4.8. *Let $\mathsf{V}^0 \subseteq \mathsf{V}$ denote the subset of nodes belonging to edges in F^0. The graph $(\mathsf{V}^0, \mathsf{F}^0)$ is connected.*

We now consider the convex hull of feasible vectors for model (**TEP**)

$$\tilde{\mathcal{P}} = \text{conv}\left(\left\{(y,x,g,\theta) \text{ satisfies } \left[\begin{array}{l} (4.1), (4.2), (4.3) \\ (4.4), (4.5) \end{array}\right. \text{ and } y \in \{0,1\}^{|\mathsf{E}^1|}\right\}\right)\,.$$

In the notation gathering parallel circuits, this means that $y_{ij}^{\ell} = 1$ for all $ij \in \mathsf{F}^0$ and each $\ell \in \mathsf{L}_{ij} = \{1, \dots, n_{ij}\}$.

For costs (4.27), the improved bound makes use of the shortest path SP_{i-j} between buses i and $j \in \mathsf{V}^0$, as well as the longest path LP^k_{i-j} between i and j, not passing through $e \in \mathsf{E}^1 \backslash \mathsf{E}^0$.

Proposition 4.9. *Suppose Assumptions 4.5 and 4.8 hold, and let ij be given. Consider constraints* (4.20) *and* (4.21) *from Section 4.3.3. Then the minimal admissible value for M_{ij} such that these constraints are valid for $\tilde{\mathcal{P}}$, for any $G, d \geq 0$, is given by*

$$\underline{M}_{ij}^{(\mathbf{TEP})} = \begin{cases} \psi_{ij} SP_{i-j} & i\,, j \in \mathsf{V}^0 \\ \psi_{ij} \max_{l \in \mathsf{V}^0} (LP^j_{i-l} + SP_{l-j}) & i \notin \mathsf{V}^0\,, j \in \mathsf{V}^0 \\ \psi_{ij} \max\left(LP_{i-j}, \max_{l_1, l_2 \in \mathsf{V}^0} (LP^j_{i-l_1} + SP_{l_1 - l_2} + LP^i_{l_2 - j})\right) & i \notin \mathsf{V}^0\,, j \notin \mathsf{V}^0\,. \end{cases}$$

Proof. When both $i, j \in \mathsf{V}^0$, the proof is similar to the one in Proposition 4.6. First, because $\psi_{ij}|\theta_i - \theta_j| \leq M_{ij}$, we must compute the maximum feasible value for the differences $|\theta_i - \theta_j|$. Once more, $\sum_{b_1 b_2 \in p} |\theta_{b_1} - \theta_{b_2}| \leq \sum_{b_1 b_2 \in p} \frac{C_{b_1 b_2}}{\psi_{b_1 b_2}}$ for any path p in F^0 between i and j. Therefore, we must have $|\theta_i - \theta_j| \leq SP_{i-j}$, since θ cannot induce flows exceeding the capacity of any existing circuit.

When $i \notin \mathsf{V}^0\,, j \in \mathsf{V}^0$, any path p from i to j must enter at least once in V^0. Let $l \in \mathsf{V}^0$ be the first entry bus and p^1 the sub-path of p from i to l. If $\sum_n y^n_{b_1 b_2} \geq 1$ for each edge $b_1 b_2 \in p^1$, then

$$|\theta_i - \theta_j| \leq \sum_{b_1 b_2 \in p^1} \frac{C_{b_1 b_2}}{\psi_{b_1 b_2}} + |\theta_l - \theta_j| \leq \sum_{b_1 b_2 \in p^1} c_{b_1 b_2} + SP_{l-j}.$$

This must be satisfied for any path p from i to j, hence, for any sub-path p_1 from j to $l \in \mathsf{V}^0$. Therefore, $|\theta_i - \theta_j| \leq \max_{l \in \mathsf{V}^0} (LP^j_{i-l} + SP_{l-j})$, as stated.

When neither i nor j belong to V^0, consider any path p from i to j. If this path does not enter in V^0, then

$$|\theta_i - \theta_j| \leq \sum_{b_1 b_2 \in p} c_{b_1 b_2}. \tag{4.31}$$

If this path crosses V^0 at least once, let l_1 be the first entry bus, l_2 be the last exit bus, p^1 the sub-path of p from i to l_1 and p^2 the sub-path of p from l_2 to j. Thus,

$$|\theta_i - \theta_j| \leq \sum_{b_1 b_2 \in p^1} c_{b_1 b_2} + |\theta_{l_1} - \theta_{l_2}| + \sum_{b_1 b_2 \in p^2} c_{b_1 b_2} \leq \sum_{b_1 b_2 \in p^1} c_{b_1 b_2} + SP_{l_1 - l_2} + \sum_{b_1 b_2 \in p^2} c_{b_1 b_2}. \tag{4.32}$$

Finally, taking the maximum of (4.31) and (4.32) over all p from i to j and considering a minimality argument, similar to the one in Proposition 4.6, ends the proof. □

We mention that a result similar to Proposition 4.9 has already been proved by Binato (2000). However, our result is more compact and general. Moreover, Theorem IV.4 from Binato (2000) contains the following (minor) glitch in equation

(79) therein. This equation states that if $ij \in \mathsf{F}^0$, then $\underline{M}_{ij}^{(\mathbf{TEP})}$ is given by C_{ij}. Actually, such statement can be made more precise when $SP_{i-j} < C_{ij}/\psi_{ij}$, because in this case $\underline{M}_{ij}^{(\mathbf{TEP})} = \psi_{ij} SP_{i-j} < C_{ij}$.

Once more, the computation of the minimal value for the third disjunctive formulation can be done as for Corollary 4.7, using the modified costs (4.27) and (4.29).

Corollary 4.10. *Suppose Assumptions 4.5 and 4.8 hold and let ij be given. Consider constraint* (4.24) *from Section 4.3.4. Then the corresponding minimal admissible value for M_{ij}^{ℓ}, for any $G, d \geq 0$, is given by*

$$\underline{M}_{ij}^{(\mathbf{TEP}),\ell} = \begin{cases} \psi_{ij} SP_{i-j} & i\,, j \in \mathsf{V}^0 \\ \psi_{ij} \max_{l \in \mathsf{V}^0}(LP_{i-l}^{j} + SP_{l-j}) & i \notin \mathsf{V}^0\,, j \in \mathsf{V}^0 \\ \psi_{ij} \max\left(LP_{i-j},\, \max_{l_1, l_2 \in \mathsf{V}^0}(LP_{i-l_1}^{j} + SP_{l_1-l_2} + LP_{l_2-j}^{i})\right) & i \notin \mathsf{V}^0\,, j \notin \mathsf{V}^0\,, \end{cases}$$

for each $\ell \in \mathsf{L}^{ij}$.

Finally, we show next how the "big-M" constraints can be further strengthened. Consider the "fat" edge $ij \in \mathsf{F}$. If circuit $(ij, 1)$ is built, $y_{ij}^1 = 1$, the difference $\psi_{ij}|\theta_i - \theta_j|$ can certainly not exceed C_{ij}, reducing M_{ij}^{ℓ} to C_{ij} for each $\ell > 1$. Hence, given that $y_{ij}^1 = 1$, we have no longer "big-M" coefficients for the remaining candidate circuits belonging to ij. Therefore, during the exploration of the branch-and-bound tree, constraints (4.33) below may yield a linear relaxation that is better than using constraints (4.20). Without loss of generality, we give the result for the standard disjunctive formulation in Section 4.3.1.

Proposition 4.11 (Improved "big-M" constraints)**.** *Suppose Assumption 4.5 holds and let ij be given. Consider constraints* (4.20) *from Section 4.3.1, and suppose symmetry is broken by using* (4.22)*. Then for all $ij \in \mathsf{F}$ and $\ell \in \mathsf{L}_{ij} \backslash \{1\}$, constraints* (4.20) *can be replaced by the reinforced constraints*

$$-(M_{ij}-C_{ij})(1-y_{ij}^1)-C_{ij}(1-y_{ij}^{\ell}) \leq x_{ij}^{\ell}-\psi_{ij}(\theta_i-\theta_j) \leq (M_{ij}-C_{ij})(1-y_{ij}^1)+C_{ij}(1-y_{ij}^{\ell})\,, \tag{4.33}$$

which are valid for $\mathcal{P}$, for any $G, d \geq 0$.

Proof. For $\ell = 1$, (4.33) is the same as (4.20). Hence, suppose $\ell > 1$. If $y_{ij}^1 = 0$, then $y_{ij}^{\ell} = 0$ for each $2 \leq \ell \leq n_{ij}+N_{ij}$, because of (4.22), and the left-(respectively, right-) most expression in (4.33) equals $-M_{ij}$(resp., M_{ij}). If $y_{ij}^1 = y_{ij}^{\ell} = 1$, then (4.33) forces $x_{ij}^{\ell} = \psi_{ij}(\theta_i - \theta_j)$. Finally, if $y_{ij}^1 = 1$ and $y_{ij}^{\ell} = 0$, (4.33) is $\psi_{ij}|\theta_i - \theta_j| \leq C_{ij}$; and constraint (4.20) for $\ell = 1$ implies that $x_{ij}^1 = \psi_{ij}(\theta_i - \theta_j)$ so that $\psi_{ij}|\theta_i - \theta_j| \leq C_{ij}$. $\square$

4.5 Computational Experiments

We make a numerical assesment comparing the different formulations from Section 4.3 on models (**TEP**) and (**TEP$_\mathbf{R}$**), with and without re-design. The main data for our test instances, based on real transmission networks, are reported in Table 4.1; for full details, we give the corresponding reference in the fourth column of the table. Note that instances "Garver", "IEEE RTS 24-bus", and "Brazil South R" allow redispatch, while the generation variables are fixed for instances "Brazil South" and "Brazil Southeast" (no redispatch is allowed).

name	Topology		Circuits		Generation/Load		References
	$\|\mathsf{V}\|$	$\|\mathsf{F}\|$	$\|\mathsf{E}^0\|$	$\|\mathsf{E}^1\|$	$\sum g$ in MW	$\sum d$ in MW	
Garver	6	15	6	90	1110	760	Alguacil et al. Garver (1970)
IEEE RTS 24-bus	24	34	38	102	10215	8560	de J. Silva Junior
Brazil South	46	79	62	237	6880	6880	Binato (2000)
Brazil South R	46	79	62	237	10545	6880	Binato (2000)
Brazil Southeast	79	143	156	429	37999	37999	Binato (2000)

Table 4.1: Networks data

The three reformulations from Section 4.3 gave the same linear relaxation for all cases from Table 4.1. In order to evaluate the impact of allowing for re-design of the network, we also compared the value of the optimal solutions for some of the models from Section 4.2. For this comparison, we used the bounds in Section 4.4 and an alternative bound, simpler to compute, that we detail next.

Remark 4.12 (Alternative lower bound). *Recall that in Section 4.4 we gave two types of lower bound for the coefficients M_{ij} in each reformulation. The first one (given by Proposition 4.9 when $i, j \in \mathsf{V}^0$) is the solution to a shortest path problem, easy to compute, which often has a small value inducing a tight linear reformulation of* (4.3) *for ij. Such is not the case for the second bound (given by Proposition 4.9 when $i \notin \mathsf{V}^0$, and by Proposition 4.6 for any $i, j \in \mathsf{V}$), because its computation requires to solve a longest path problem.*

Indeed, being a generalization of the Traveling Salesman Problem, *the longest path problem itself is very difficult to solve (Hardgrave and Nemhauser, 1962). Although polynomial algorithms have been proposed for special classes of graphs, see for instance Uno (2007), solving the problem for general graphs requires to develop a specialized branch-and-cut algorithm, which is beyond the scope of this work. Moreover, since the value of the second bound, $\underline{M}_{ij}$, is already very big, considering an alternative bigger bound neither modifies the quality of the linear relaxation nor decreases the solution times. This remark was confirmed by a set of unreported tests, with increasing values for coefficients M_{ij}. Therefore, instead of trying to solve a longest path problem, we use the following alternative upper bound $\underline{MM}_{ij}$ for $\underline{M}_{ij}$, with value depending on the considered model:*

Model ($\mathbf{TEP_R}$) *with re-design: For each bus $b \in \mathsf{V}$, let $f_b \in \mathsf{F}$ be the maximum-cost "fat" edge connected to b, according to costs* (4.27). *Summing up the costs $\tilde{c}_{f_b}$ we certainly get an upper bound on the length of any path in the graph*

(V,F). *We can reduce this value, by forbidding to pick up twice the same edge. After ordering the buses* $b_1,\ldots,b_{|\mathsf{V}|}$, *this is formally written as*

$$\underline{MM}_{ij}^{(\mathbf{TEP_R})} = \psi_{ij}\sum_{i=1}^{|\mathsf{V}|}\max_{b_ib_j\in\mathsf{F}_{b_i}}\tilde{c}_{b_1b_2}, \tag{4.34}$$

where $\mathsf{F}_{b_i}=\mathsf{F}\backslash\{f_{b_1},\ldots,f_{b_{i-1}}\}$ *for* $i=2,\ldots,|\mathsf{V}|$.

Classical transmission expansion planning model ($\mathbf{TEP}$*): When an initial structure* $(\mathsf{V}^0,\mathsf{F}^0)$ *is given and* $i\notin\mathsf{V}^0$, *the bound* (4.34) *can be reduced with no additional computational effort to*

$$\underline{MM}_{ij}^{(\mathbf{TEP})} = \psi_{ij}\left(\max_{b_1,b_2\in\mathsf{V}^0}SP_{b_1-b_2}+\sum_{i=1}^{|\mathsf{V}\backslash\mathsf{V}^0|}\max_{b_ib_j\in\mathsf{F}^1_{b_i}}\tilde{c}_{b_1b_2}\right),$$

where $\mathsf{F}^1_{b_i}=\mathsf{F}^1\backslash\{f_{b_1},\ldots,f_{b_{i-1}}\}$ *for* $i=2,\ldots,|\mathsf{V}|$.

All the codes were written in JAVA, using CPLEX concert technology 11 (IBM-ILOG, 2009), on a computer with an Intel Core 2 Duo processor at 2.40 GHz and 2 GB of RAM memory.

Although we used the MIP black-box solver of CPLEX to handle both (**TEP**) and ($\mathbf{TEP_R}$), we provided CPLEX with a lower bound for (**TEP**), and lower and upper bounds for ($\mathbf{TEP_R}$). The lower bound was obtained from the optimal value t of Garvers's transportation model: we added the constraint $\sum_{e\in\mathsf{E}^1}c_ey_e \geq t$ to the formulations. Then, solving first (**TEP**), we could provide CPLEX with a starting feasible solution defining an upper bound. Both bounds reduced significantly the solution times.

Tables 4.2, 4.3, and 4.4 report the results obtained for each formulation from Section 4.3, for models (**TEP**), ($\mathbf{TEP_R}$), and (**Transportation**), respectively.

name	Standard		Improved		Ordering		Alternative	
	T	nodes	T	nodes	T	nodes	T	nodes
Garver	0.015	5	0.078	8	0.093	11	0.031	45
IEEE	0.23	81	0.92	363	0.56	121	0.67	499
South	5.085	4403	47.75	12149	24.18	5595	48.61	31768
South R	1.51	579	9.079	1823	1.88	395	3.52	1508
Southeast	2438	468428	2888	317314	599	113460	5095	2096088

Table 4.2: Results for the formulations from Section 4.3 on (**TEP**).

In each table, columns "Standard", "Improved", and "Alternative" stand for formulations from sections 4.3.1, 4.3.2, and 4.3.4, respectively, whereas "Ordering" refers to the "Standard" formulation with the addition of constraints (4.22), and "Reinforced" refers to "Ordering", using constraints (4.33) instead of (4.20). Solution times are given in seconds, and they exclude the time needed to compute the bounds. We also give the number of nodes explored in the branch-and-bound trees for the different formulations.

name	Standard		Ordering		Reinforced	
	T	nodes	T	nodes	T	nodes
Garver	0.124	0	0.0936	0	0.0468	0
IEEE	50	12756	13	5021	12	3999
South	35091	1368186	19052	952880	20438	1079935
South R	144	14933	31	4408	29	3155

Table 4.3: Results for the formulations from Section 4.3 on (**TEP$_\mathbf{R}$**).

Note that Table 4.3 does not contain the results for "Brazil Southeast" network, because none of the formulations could solve that instance within 10 hours of computing time. For such a large network, allowing for re-design with the formulations from Section 4.3 would require to develop a more sophisticated branch-and-cut algorithm.

name	T	nodes
Garver	0.015	0
IEEE	0.031	14
South	2.71	1351
South R	0.078	57
Southeast	0.343	137

Table 4.4: Model (**Transportation**).

Table 4.5 contains the optimal values found for the models from Sections 4.2.1, 4.2.2, and (**Transportation**). Columns "LPrelax" report the values of the LP relaxations at the root node and the rounded gaps $\frac{Optimal - LPrelax}{Optimal}$. For small instances, reinforced constraints (4.33) do not improve the solutions times. However, when used in conjunction with a high branching priority for y^1_{ij}, we could obtain a better upper bound for model (**TEP$_\mathbf{R}$**) of network "Brazil Southeast" (the same branching strategy applied to smaller instances increased the solution times). Apart from this special case, all parameters of CPLEX were left to their default values.

Our results from Table 4.5 for (**TEP**) coincide with the best ones reported in the literature. In this sense, any cost below these values can be considered as an improvement. We see on Table 4.5 that the (**TEP$_\mathbf{R}$**) model for "Brazil South" network induces a cost reduction of $\mathbf{9.67} = 72.87 - 63.2$ and $\mathbf{8.2} = 154.4 - 146.2$ for the cases with and without redispatch, respectively. For "Brazil Southeast", the

name	(**TEP**)		(**TEP**$_{\mathbf{R}}$)		(**Transportation**)	
	Optimal	LPrelax	Optimal	LPrelax	Optimal	LPrelax
Garver	110	99 – 10%	110	99 – 10%	110	99 – 10%
IEEE	152	75 – 50%	152	68.8 – 55%	102	69 – 32 %
South	154.4	82 – 47%	146.2	71.8 – 51%	127	72 – 43%
South R	72.87	41 – 44%	63.2	33 – 48%	53	33 – 38%
Southeast	424.8	173 – 59%	$\leq$ 405.9	120 – N.A.	284	120 – 58%

Table 4.5: Optimal costs and relaxations.

best cost obtained was 405.9 after 10 hours of computing time, with a duality gap of about 29% (the best cost obtained when using (4.20) instead of (4.33) was 412). Thus, also for this network, when comparing with the best known values, we have already a cost reduction of **18.9** = 424.8 – 405.9. At least on our tested instances, allowing the network to be re-designed can bring important savings in transmission expansion investments.

4.6 Towards (N-1) reliability

In the previous sections, we investigated some of the difficulties of the transmission expansion problem, leaving aside an important criterion: the designed network must be resistant against the failure of any of its circuits. In principle, $(N-1)$ reliability constraints should ensure that the network remains operational if any of its circuits happened to fail alone. However, in view of (4.6), it may be too restrictive (thus, too expensive) to require the whole load to be supplied under any circuit failure. This is especially true if the failure of some link would prevent the network from working, whereas not attending to a small portion of the load while the circuit is repaired would keep the network operational.

Therefore, we model the problem as a two-stage stochastic program with continuous recourse (Birge and Louveaux, 2008). First stage variables y_e indicate which circuits are built (or left operational, when $e \in \mathsf{E}^0$). At the second stage, for each contingency scenario $h \in \mathsf{E}$, we define continuous shortage variables u_i^h, $i \in \mathsf{V}$, and associated binary coefficients δ_e^h stating which circuits are operational:

$$\delta_e^h = \begin{cases} 0 & \text{if } h = e \\ 1 & \text{if } h \neq e. \end{cases}$$

In order to take into account the fact that the network must supply the whole load when all of its circuits are operational, we also define the scenario "*all*" (and $\mathsf{E}^* = \mathsf{E} \cup \{all\}$), corresponding to $u^{all} \equiv 0$ and $\delta^{all} \equiv 1$. Finally, the vector $(x^h, g^h, u^h, \theta^h)$ describes the routing for each scenario $h \in \mathsf{E}^*$.

With these additional variables, the $(N-1)$ criterion for $(\mathbf{TEP_R})$ has the form:

$$
(\mathbf{TEP_R} - \mathbf{N1}) \quad
\begin{array}{lll}
\min & \displaystyle\sum_{e \in \mathsf{E}^1} c_e y_e + \sum_{h \in \mathsf{E}, b \in \mathsf{V}} p_i^h u_i^h & \\
\text{s.t.} & \displaystyle\sum_{e \in \mathsf{E}: t(e)=i} x_e^h - \sum_{e \in \mathsf{E}: s(e)=i} x_e^h = d_i - g_i^h - u_i^h & i \in \mathsf{V},\, h \in \mathsf{E}^* \\
& x_e^h - \psi_e \delta_e^h y_e (\theta_{s(e)}^h - \theta_{t(e)}^h) = 0 & e \in \mathsf{E},\, h \in \mathsf{E}^* \quad (4.35) \\
& -C_e \le x_e^h \le C_e & e \in \mathsf{E},\, h \in \mathsf{E}^* \quad (4.36) \\
& 0 \le g_i^h \le G_i & i \in \mathsf{V},\, h \in \mathsf{E}^* \\
& 0 \le u_i^h \le d_i & i \in \mathsf{V},\, h \in \mathsf{E}^* \\
& u_i^{all} = 0 & i \in \mathsf{V} \\
& y \text{ binary} &
\end{array}
$$

In the objective function above, each penalty factor p_i^h is an estimation of the practical cost of shortage for bus i, multiplied by the probability of failure h to happen. Then, we can use any of the linearized reformulations from Section 4.3 to handle the bilinear constraints. For instance, with the standard approach from Section 4.3.1, we replace constraints (4.35) by

$$
-M_e(1 - \delta_e^h y_e) \le x_e^h - \psi_e(\theta_{s(e)}^h - \theta_{t(e)}^h) \le M_e(1 - \delta_e^h y_e), \qquad \text{for all } e \in \mathsf{E},\, h \in \mathsf{E}^*, \tag{4.37}
$$

and constraints (4.36) by

$$
|x_e^h| \le \delta_e^h y_e C_e \qquad \text{for all } e \in \mathsf{E},\, h \in \mathsf{E}^*, \tag{4.38}
$$

yielding the model introduced by P. Tsamasphyrou and Carpentier (1999) (without re-design).

We mention that Oliveira et al. (2007) also make a two-stage formulation of the $(N-1)$ criterion. However, their model considers a different bus-circuit incidence matrix (matrix from equations (4.1)) for each scenario $h \in \mathsf{E}^*$. Such second-stage matrices are defined by suppressing the column related to circuit h. Our simpler recourse formulation $(\mathbf{TEP_R} - \mathbf{N1})$, with a fixed bus-circuit incidence matrix, should ease the use of Stochastic Programming decomposition algorithms.

Model	time	nodes	optimal	LPrelax
(**Transportation**)	0.5	11	116	106.7 – 8%
(**TEPN1**)	3.8	31	118.4	106.7 – 9.8%
$(\mathbf{TEP_R} - \mathbf{N1})$	8.3	67	118.4	106.7 – 9.8%

Table 4.6: $(N-1)$ reliability constraints for Garver's network.

We can see preliminary computational results on Table 4.6 for Garver's network, using again the MIP solver of CPLEX 11. Model (**TEPN1**) stands for ($\mathbf{TEP_R} - \mathbf{N1}$) with additional constraints $y_e = 1$ for $e \in \mathsf{E}^0$, and "Transportation" for ($\mathbf{TEP_R} - \mathbf{N1}$) without (Kirchoff). Since the considered network is small, there is no "slack" for the re-design model to give any improvement: the optimal values of (**TEPN1**) and ($\mathbf{TEP_R} - \mathbf{N1}$) are equal. For this reason, rather than giving insight on the model with re-design, our results in Table 4.6 should be considered as a validation of our solving methodology for ($\mathbf{TEP_R} - \mathbf{N1}$).

Part II

Stochastic models

5

The stochastic knapsack problem with simple recourse

5.1 Introduction

One of the difficulties of network design problems comes from their capacity constraints. Many other mixed-integer programs feature these constraints such as capacitated facility locations and lot-sizing problems. Yet, the knapsack problem is one the simplest problem with a capacity constraint. The study of the knapsack problem has proven to be useful for a better understanding of the capacity constraint, yielding strong cutting planes, among which the cover cuts. These cuts are able to strengthen the linear relaxation of most problems featuring a capacity constraint with binary variables. For instance, it has been used successfully for the unsplittable (see Chapter 7) multi-commodity flow problem by Barnhart et al. (2000). Similarly, the study of the robust knapsack problem, where weights belong to a polyhedron $\mathcal{W}$, yielded robust cover cuts which have been used successfully for the robust bandwidth packing problem (Klopfenstein and Nace). Alternatively, knapsack problems appear within decomposition schemes for more general problems with capacity constraints. For instance, efficient algorithms for the generalized assignment problem rely on a Lagrangian relaxation of some of the constraints, yielding a couple of knapsack problems (Nauss, 2003).

The examples above motivate the study of the stochastic knapsack problem. This chapter shows of some of the properties of the stochastic knapsack problem with simple recourse. In Chapter 6, we present an extension of the (numerical) methodology studied herein to the stochastic network design problem with simple recourse.

Given a set of items N, the classical knapsack problem looks for a subset of N, whose total weight does not exceed the capacity C of the knapsack and which maximizes the total profit. Each item $i \in \mathsf{N}$ has a profit p_i and a weight w_i. Our two-stage stochastic version of the problem considers that weights $w_i(\omega)$ and capacity

$C(\omega)$ are random variables. The subset of items is chosen first, then any real amount $z(\omega)$ of additional capacity can be bought at the unit price of K, depending on the scenario ω. The objective function maximizes the profit of the chosen subset minus the expected cost of the additional capacity:

$$\textbf{(KSR)} \qquad \begin{array}{lll} \max & \sum_{i \in \mathsf{N}} p_i x_i - K\mathbb{E}_\xi[z(\omega)] & \\ \text{s.t.} & \sum_{i \in \mathsf{N}} w_i(\omega) x_i \leq C(\omega) + z(\omega) & \forall \omega \in \Omega \qquad (5.1) \\ & x \text{ binary}, z(\omega) \geq 0, & \end{array}$$

where Ω is the set of all scenarii and $\mathbb{E}_\xi[z(\omega)] = \int_\Xi z(v) dF_\xi(v)$ where $F_\xi : \mathbb{R}^{n+1} \to \mathbb{R}_+$ is the distribution function of the random vector $\xi = (C, w_1, \ldots, w_n) : \Omega \to \Xi \subset \mathbb{R}^{n+1}$. Because constraints (5.1) are satisfied at equality in any optimal solution with $z(\omega) > 0$, variables $z(\omega)$ can be replaced by $\max(0, \sum w_i(\omega) x_i - C(\omega))$, resulting in the following formulation:

$$\max_{x \in \{0,1\}^n} \sum_{i \in \mathsf{N}} p_i x_i - K\mathbb{E}_\xi \left[\max \left(0, \sum_{i \in \mathsf{N}} w_i(\omega) x_i - C(\omega) \right) \right]. \qquad (5.2)$$

This problem can be thought of as the following resource allocation problem (Kleywegt et al., 2002). A decision maker has to choose a subset of n known alternative projects to take on. For this purpose a known quantity C of relatively low-cost resource is available to be allocated. Any additional amount of resource required can be obtained at a known incremental cost of K per unit of resource. The amount w_i of resource required by each project is not known at the time the decision has to be made, but we assume that the decision maker has an estimate of the probability distribution of those w_i. Finally, each project i has an expected reward of p_i.

Besides being of practical application, the expected term in the objective makes the stochastic model worth to be studied. Indeed, a well known simplification when dealing with stochastic models is to replace random parameters by their means and to solve the resulting deterministic model. However, the term

$$\mathbb{E}_\xi \left[\max \left(0, \sum_{i \in \mathsf{N}} w_i(\omega) x_i - C(\omega) \right) \right]$$

implies that two subsets N_1 and N_2 having equal means but different variances, may have different expected cost. This is not taken into account by the deterministic model considering only means.

Problem (**KSR**) has been first formulated by Cohn and Barnhart (1998) who consider that C is given and that w_i follow Gaussian variables. They derive basic properties and propose a simple branch-and-bound algorithm that they test on an example with 15 variables. Recently, Kosuch and Lisser (2009) use a stochastic gradient method to solve (**KSR**) with Gaussian variables. They solve the problem with up to 150 variables in at most two hours. Kleywegt et al. (2002) work on a similar model with discrete random variables to test their sample average approximation method. Other ways of considering uncertainties in the parameters of the knapsack problem include chance-constrained knapsack (Klopfenstein and Nace, 2008; Morton and Wood, 1998), robust knapsack (Klopfenstein and Nace, 2007) and dynamic knapsack (Kleywegt and Papastavrou, 2001), among others.

Even though problem (5.2) is not linear, it is unconstrained and concave as explained below. Let the expected optimal value of the second stage problem be denoted by the recourse function $\mathcal{Q}$. We say that the program has a fixed recourse if the constraint coefficients of the recourse variables are constant.

Theorem 5.1 (Birge and Louveaux (2008)). *For a maximization stochastic program with fixed recourse, $\mathcal{Q}(x)$ is a concave function on the convex set $K_2 = \{x|\mathcal{Q}(x) < \infty\}$.*

Coming back to (**KSR**), the recourse function is defined by

$$\mathcal{Q}(x) = -K\mathbb{E}_\xi \left[\min_{z(\omega)\geq 0} \left\{ z(\omega) \text{ s.t. } \sum_{i\in \mathsf{N}} w_i(\omega)x_i \leq C(\omega) + z(\omega) \right\} \right],$$

and the only recourse variable is z with a coefficient equal to 1; moreover $K_2 = \mathbb{R}^n$. Hence we can think of using convex non-linear mixed integer techniques to tackle the problem. For instance outer-approximation type algorithms (Duran and Grossmann, 1986) approach the objective by a set of tangent planes. However to compute the coefficients of those planes, we need to evaluate many times the recourse function

$$\begin{aligned} \mathcal{Q}(x) &= -K\mathbb{E}_\xi \left[\max\left(0, \sum_{i\in \mathsf{N}} w_i(\omega)x_i - C(\omega)\right)\right] \\ &= -K \int_\Xi \max\left(0, \sum w_i x_i - C\right) dF_\xi(C, w_1, \dots, w_n), \end{aligned} \tag{5.3}$$

namely to evaluate an integral of n variables which requires to use numerical integration packages. Alternatively, we may replace the random vector, in this case $\xi = (C, w_1, \dots, w_n)$, by a discrete approximation yielding a discrete set of scenarii. This results in solving a large scale problem, with a particular structure well suited for Benders' decomposition methods such as the ones described in Chapter 3, also called L-shaped (Laporte et al., 2002). When the number of scenarii is large, some authors use Monte Carlo sampling to generate a few scenarii instead of considering all of them (Kleywegt et al., 2002; Sen et al., 1994).

The contributions of this chapter are twofold. First, we prove that three special cases of the stochastic knapsack problem are weakly $\mathcal{NP}$-hard:

- Stochastic knapsack with known weights and uniformly distributed capacity.
- Stochastic subset sum with Gaussian weights and positive random capacity.

- Stochastic subset sum with known weights and arbitrary random capacity.

Then, we show that the LP/NLP algorithm of Quesada and Grossman (1992) allows us to solve efficiently the general problem with Gaussian random variables and provide computational results. The implemented algorithm is able to solve in less than a minute problems involving up to a few thousands of variables.

The chapter is organized as follows. Next section studies the complexity of the special cases of the stochastic knapsack problem, prescribing pseudo-polynomial algorithms. Section 5.3 presents an algorithm for the general models and provide extensive computational experiments. A comparison of the general algorithm and the pseudo-polynomial approaches is also presented for the special cases.

5.2 Pseudo-polynomial cases

In this section we show that Problem (**KSR**) is in general weakly $\mathcal{NP}$-hard. We then turn to special cases of (**KSR**) that can be solved in pseudo-polynomial time.

Proposition 5.2. *Problem* (**KSR**) *is at least as hard as the classical knapsack problem.*

Proof. Consider (**KSR**) for a unique scenario. Writing the capacity constraint explicitly, the problem reads:

$$\max_{x \in \mathsf{B}, z \geq 0} \left\{ \sum_{i \in \mathsf{N}} p_i x_i - Kz \text{ s.t. } \sum_{i \in \mathsf{N}} w_i x_i \leq C + z \right\}. \tag{5.4}$$

Taking K large enough, z is equal to 0 in any solution so that (5.4) is equivalent to the classical knapsack problem. □

In the rest of this section, we use the notations B and $\mathcal{B}$ to denote $\{0,1\}^n$ and $[0,1]^n$, respectively; the summation $\sum$ refers to the sum over N unless stated otherwise.

5.2.1 Fixed weights and uniformly distributed capacity

In this section, we consider that weights w_i are fixed so that (**KSR**) becomes:

$$\max_{x \in \mathsf{B}} \sum p_i x_i - K \int_0^{\sum w_i x_i} \left(\sum w_i x_i - C \right) dF(C). \tag{5.5}$$

Assuming that C is uniformly distributed between positive integers $\underline{C}$ and $\overline{C}$ and that all parameters are integers we show next that the optimization problem can be solved by a pseudo-polynomial algorithm under a mild assumption on its parameters. Note that this problem can also be seen as a robust knapsack problem with linear penalty (Mulvey et al., 1995). With C uniformly distributed, (5.5) becomes:

$$\max_{x \in \mathsf{B}} \sum p_i x_i - \frac{K}{\overline{C} - \underline{C}} \int_{\min(\sum w_i x_i, \underline{C})}^{\min(\sum w_i x_i, \overline{C})} \left(\sum w_i x_i - C \right) dC, \tag{5.6}$$

where we assume $\sum w_i > \underline{C}$ to avoid the trivial solution $x = (1, \ldots, 1)$.

Theorem 5.3. *Problem* (5.6) *is in general $\mathcal{NP}$-complete. However, if* $\left(p_i + \frac{Kw_i}{2(\overline{C}-\underline{C})}(2\underline{C} - w_i)\right) \geq 0$ *for each* $i \in \mathsf{N}$*, the problem can be solved in* $O(nK \sum w_i)$.

For a large number of items with individual volumes small enough, this condition is likely to be satisfied. Applying an argument similar to Proposition 5.2 to a problem with $\overline{C} = \underline{C} + 1$, we obtain:

Lemma 5.4. *Problem* (5.6) *is at least as hard as its deterministic counterpart.*

The rest of the section show how (5.6) can be solved in pseudo-polynomial time whenever $\left(p_i + \frac{Kw_i}{2(\overline{C}-\underline{C})}(2\underline{C} - w_i)\right) \geq 0$ for each $i \in \mathsf{N}$. Let Z be the the function to be maximized in (5.6). We rewrite Z as follows:

$$Z(x) = \begin{cases} Z_1(x) = & \sum p_i x_i & \textit{for} \ \ \sum w_i x_i \leq \underline{C} \\ Z_2(x) = & \sum p_i x_i - \frac{K}{2(\overline{C}-\underline{C})}\left(\sum w_i x_i - \underline{C}\right)^2 & \textit{for} \ \ \underline{C} \leq \sum w_i x_i \leq \overline{C} \\ Z_3(x) = & K\frac{\overline{C}+\underline{C}}{2} + \sum (p_i - Kw_i) x_i & \textit{for} \ \ \overline{C} \leq \sum w_i x_i. \end{cases} \tag{5.7}$$

Lemma 5.5. *Let* x^*, x_1^*, x_2^* *and* x_3^* *be optimal solutions of* $\max_{\mathsf{B}} Z(x)$, $\max_{\mathsf{B}}\{Z_1(x)$ *s.t.* $\sum w_i x_i \leq \underline{C}\}$, $\max_{\mathsf{B}} Z_2(x)$ *and* $\max_{\mathsf{B}}\{Z_3(x)$ *s.t.* $\sum w_i x_i \geq \overline{C}\}$, *respectively. Then,* $Z(x^*) = \max(Z_1(x_1^*), Z_2(x_2^*), Z_3(x_3^*))$.

Proof. We can relax the domain restriction $\underline{C} \leq \sum w_i x_i \leq \overline{C}$ for Z_2 because $Z_1(x) \geq Z_2(x)$ and $Z_3(x) \geq Z_2(x)$ for any $x \in \mathcal{B}$. □

The following three lemmas show that each of the problems from Lemma 5.5 can be solved in pseudo-polynomial time, this proving Theorem 5.3.

Lemma 5.6. *Maximizing* $Z_1(x)$*, for* x *binary and* $\sum w_i x_i \leq \underline{C}$*, can be done in* $O(n \sum w_i)$.

Proof. This is a knapsack problem, which can be optimized in $O(n\underline{C})$ and thus in $O(n \sum w_i)$ because $\underline{C} < \sum w_i$. □

Lemma 5.7. *Maximizing* $Z_3(x)$*, for* x *binary and* $\sum w_i x_i \geq \overline{C}$*, can be done in* $O(n \sum w_i)$.

Proof. In the following, we assume that $\sum w_i > \overline{C}$, otherwise $\sum w_i x_i$ is always smaller than $\overline{C}$ so that the problem does not have a solution. We show that the problem is a knapsack problem.

i. Define $\mathsf{M} = \{i \in \mathsf{N} \mid p_i - Kw_i < 0\}$ and $\tilde{C} = \sum w_i - \overline{C}$; let x^* be the solution to the following knapsack problem ($x_i^* = 0$ for $i \in \mathsf{N}/\mathsf{M}$)

$$\max_{\mathsf{B}} \left(\sum_{i \in \mathsf{M}} (Kw_i - p_i) x_i \quad \text{s.t} \quad \sum_{i \in \mathsf{M}} w_i x_i \leq \tilde{C} \right). \tag{5.8}$$

ii. An optimal solution to $\max_{\mathsf{B}}\{Z_3(x) \text{ s.t. } \sum w_i x_i \geq \overline{C}\}$ is given by $x_i = 1 - x_i^*$, for each $i \in \mathsf{N}$.

Then, because $\tilde{C} \leq \sum w_i$, (5.8) can be solved in $O(n \sum w_i)$. □

Lemma 5.8. *If* $\left(p_i + \frac{Kw_i}{2(\overline{C}-\underline{C})}(2\underline{C} - w_i)\right) \geq 0$ *for each* $i \in \mathsf{N}$*, maximizing* $Z_2(x)$*, for* x *binary, can be done in* $O(nK \sum w_i)$*.*

Proof. Expanding the square in Z_2, and using the identity $x_i^2 = x_i$ because $x \in \mathsf{B}$, we obtain:

$$-\frac{K\underline{C}^2}{2(\overline{C}-\underline{C})} + \max_{x \in \mathsf{B}} \left(\sum_{i \in \mathsf{N}} \left(p_i + \frac{Kw_i}{2(\overline{C}-\underline{C})}(2\underline{C} - w_i)\right) x_i - \frac{K}{\overline{C}-\underline{C}} \sum_{\substack{i,j \in \mathsf{N} \\ i \neq j}} w_i w_j x_i x_j \right). \tag{5.9}$$

Assuming that $\left(p_i + \frac{Kw_i}{2(\overline{C}-\underline{C})}(2\underline{C} - w_i)\right) \geq 0$ for each $i \in \mathsf{N}$, linear coefficients of (5.9) are all positive. We need them to be integer as well to apply the tools from optimization of pseudo boolean functions.

Multiplying all terms by $4K(\overline{C} - \underline{C})$, (5.9) becomes a particular case of half-products (Badics and Boros, 1998)

$$f = \sum_{i \in \mathsf{N}} c_i x_i - \sum_{\substack{i,j \in \mathsf{N} \\ i \neq j}} a_i b_j x_i x_j,$$

where $c \mapsto 4K(\overline{C}-\underline{C})\left(p + \frac{Kw}{2(\overline{C}-\underline{C})}(2\underline{C} - w_i)\right)$ and $a = b \mapsto 2Kw$. Badics and Boros (1998) provide a dynamic programming algorithm for general half-products with positive coefficients. Its running time is $O(nA)$, where $A = 2K \sum w_i$. □

Besides this dynamic programming approach, new versions of optimization softwares, including CPLEX 11, can manage maximization of integer problems with a concave and quadratic objective function. Nevertheless, we show in Section 5.3.2 that the LP/NLP algorithm described in the Section 5.3.1 solves (5.9) much faster than do CPLEX 11 and the algorithm from Badics and Boros.

5.2.2 Subset sum

The subset sum problem is a well known particular case of the deterministic knapsack problem is that assumes that weight w_i is equal to profit p_i for each item $i \in \mathsf{N}$. Even though weakly $\mathcal{NP}$-hard to solve, adapted algorithms can have a much better behavior for this problem than for the general knapsack.

To the best of our knowledge, no stochastic version of the subset sum has been addressed in the literature so far. To define the stochastic subset sum, we replace the deterministic constraint $w = p$ by:

- $\mathbb{E}_\xi[w] = p$,
- $\mathrm{Var}_\xi[w] = \lambda p$ for some $\lambda \geq 0$,

where $\mathbb{E}_\xi[v] = (\mathbb{E}_\xi[v_1], \dots, \mathbb{E}_\xi[v_n])$ and $\mathrm{Var}_\xi[v] = (\mathrm{Var}_\xi[v_1], \dots, \mathrm{Var}_\xi[v_n])$ for any random vector v. The constraint $\mathbb{E}_\xi[w] = p$ is the direct extension of $w = p$. Then, to enforce the link between p and w we also impose $\mathrm{Var}_\xi[w] = \lambda p$. Note that the case $\lambda = 0$ results in a deterministic knapsack, where additional capacity can be purchased at the incremental cost of K per unit, see problem (5.4).

Gaussian weights

In this section, we assume that weights w_i, $i \in \mathsf{N}$, are independent Gaussian variables with parameters μ_i and $\sigma_i^2 = \lambda\mu_i$, $i \in \mathsf{N}$, for some $\lambda \geq 0$, and that capacity C is a positive random variable. This is motivated by the following summation property: if $w_1, \dots, w_n$ are independent Gaussians of mean μ_i and variance σ_i^2, and x_i are real numbers, then $Y := \sum x_i w_i \sim \mathcal{N}(\mu(x), \sigma^2(x))$, with $\mu(x) = \sum x_i \mu_i$ and $\sigma^2(x) = \sum x_i^2 \sigma_i^2$. Moreover, Gaussians are often used to represent the error made on estimations of parameters for many physical and economical problems.

Theorem 5.9. *Consider the problem*

$$\max_{x \in \mathsf{B}} \sum \mu_i x_i - K\mathbb{E}_\xi \left[\max\left(0, \sum w_i(\omega)x_i - C(\omega)\right)\right], \tag{5.10}$$

where $w_i \sim \mathcal{N}(\mu_i, \lambda\mu_i)$, $0 \leq \lambda \leq 1$, μ is an integer vector and C is a positive random variable. Problem (5.10) *is weakly $\mathcal{NP}$-hard and can be solved by a pseudo-polynomial algorithm in $O(n \sum \mu_i)$.*

The fact that (5.10) is at least weakly $\mathcal{NP}$-hard easily follows from an argument similar to the one used in Proposition 5.2, taking $\lambda = 0$. The rest of the section shows how to construct a pseudo-polynomial algorithm for (5.10). Let $w(x) = \sum w_i x_i \sim \mathcal{N}(\mu(x), \sigma^2(x))$, so that the usual recourse function reads

$$\mathcal{Q}(x) = -K\mathbb{E}_\xi[\max(0, w(x;\omega) - C(\omega))],$$

and consider the auxiliary function

$$\mathcal{R}(x) = -K\mathbb{E}_\xi[\max(0, \hat{w}(x;\omega) - C(\omega))],$$

where $\hat{w}(x) \sim \mathcal{N}(\mu(x), \hat{\sigma}^2(x))$, with $\mu(x) = \sum \mu_i x_i$ as before and $\hat{\sigma}^2(x) = \sum \sigma_i^2 x_i$. Note that for each $i \in \mathsf{N}$, $x_i^2 = x_i$ when $x \in \mathsf{B}$ so that $\mathcal{Q}(x) = \mathcal{R}(x)$ when $x \in \mathsf{B}$.

We define then $Z_\mathcal{Q}(x) = \sum \mu_i x_i + \mathcal{Q}(x)$ and $Z_\mathcal{R}(x) = \sum \mu_i x_i + \mathcal{R}(x)$, so that functions $Z_\mathcal{Q}(x)$ and $Z_\mathcal{R}(x)$ coincide on B, and $\max_\mathsf{B} Z_\mathcal{Q}(x) = \max_\mathsf{B} Z_\mathcal{R}(x)$. In what follows, we focus on the maximization of $Z_\mathcal{R}$. This is motivated by the following property:

Lemma 5.10. *If $\sigma_i^2 = \lambda\mu_i$ for each $1 \leq i \leq n$ and some $\lambda \geq 0$, then there exists a function $\hat{Z} : [0, \sum \mu_i] \to \mathbb{R}$ such that for all $x \in \mathcal{B}$, $Z_\mathcal{R}(x) = \hat{Z}(z)$ with $z = \sum \mu_i x_i$.*

Proof. By definition of z, $Z_{\mathcal{R}}$ can be rewritten as $z+\mathcal{R}(x)$. Then, we see that $\mu(x) = \sum \mu_i x_i = z$ and $\hat{\sigma}(x) = \sqrt{\sum \sigma_i^2 x_i} = \sqrt{\lambda \sum \mu_i x_i} = \sqrt{\lambda z}$ proving the result. □

Notice that $\mathcal{R}(x) = \mathcal{Q}(x)$ only when $x \in \mathsf{B}$; when $x \in \mathcal{B}/\mathsf{B}$ these functions may be different. In particular, neither $Z_{\mathcal{R}}$ nor $\hat{Z}$ inherit from the concavity of Theorem 5.1. Nevertheless, we can prove the result analytically for $\hat{Z}$:

Lemma 5.11. *If* $0 < \lambda \leq 1$ *and* C *takes positive values only, the function* $\hat{Z}$ *is concave on its domain.*

Proof. Let f and F_C be the density function of $\mathcal{N}(0,1)$ and the distribution function of C, respectively. $\hat{Z}$ is defined by the following expression

$$\hat{Z}(z) = z - K \int_0^\infty \left\{ \frac{1}{\lambda z} \int_0^\infty f\left(\frac{w - z + C}{\lambda z}\right) dw \right\} dF(C). \tag{5.11}$$

Following Cohn and Barnhart (1998), among others, the inner integral can be simplified, yielding:

$$\hat{Z}(z) = z - K \int_0^\infty \left\{ \lambda z f\left(\frac{C-z}{\lambda z}\right) + (z - C) G\left(\frac{C-z}{\lambda z}\right) \right\} dF_C(C), \tag{5.12}$$

where $G = 1 - \Phi$ and Φ is the distribution function of $\mathcal{N}(0,1)$. Computing the second derivative of (5.12) for any $z > 0$, we obtain:

$$\hat{Z}''(z) = -K \int_0^\infty \left\{ \frac{\lambda^{-2}(z+C)^2 - z}{8\lambda\sqrt{\pi}\sqrt{z}^5} e^{-\frac{(z-C)^2}{2\lambda^2 z}} \right\} dF_C(C). \tag{5.13}$$

The integrand of (5.13) is non-negative when $C \geq 0$ and F_C is equal to 0 otherwise, so that $\hat{Z}''(z)$ is non-positive for all $z > 0$. □

Recalling that $\sigma^2 = \lambda\mu$, the assumption $\lambda \leq 1$ becomes $\sigma^2 \leq \mu$. We see in Section 5.3.1 that this assumption is required if we want $P(w_i \leq 0)$ to be negligible. Hence, in the following we always assume $\sigma^2 \leq \mu$, so that the function $\hat{Z}$ is concave.

Because $\hat{Z}$ is concave, it has at most one maximum. Suppose that we can compute the maximum z^* of $\hat{Z}$ over $\mathbb{R}^+$, which may be greater than $\sum \mu_i$. The concavity of $\hat{Z}$ implies that

$$z_1 \leq z_2 \leq z^* \Rightarrow \hat{Z}(z_1) \leq \hat{Z}(z_2) \qquad \text{and} \qquad z^* \leq z_2 \leq z_1 \Rightarrow \hat{Z}(z_1) \leq \hat{Z}(z_2),$$

for any $z_1, z_2 \in [1, \sum \mu_i]$. Recalling that $\hat{Z}(\sum \mu_i x_i) = Z_{\mathcal{R}}(x)$, we can write similar inequalities for $Z_{\mathcal{R}}$:

$$\begin{aligned} &\sum \mu_i x_{1i} \leq \sum \mu_i x_{2i} \leq z^* \Rightarrow Z_{\mathcal{R}}(x_1) \leq Z_{\mathcal{R}}(x_2) \\ \text{and} \qquad &z^* \leq \sum \mu_i x_{2i} \leq \sum \mu_i x_{1i} \Rightarrow Z_{\mathcal{R}}(x_1) \leq Z_{\mathcal{R}}(x_2), \end{aligned}$$

for any fractional vectors $x_1, x_2 \in \mathcal{B}$.

Hence, the closer $\sum \mu_i x_i$ is o z^*, the higher is $Z_{\mathcal{R}}(x)$. Then, two situations can happen. If $\sum \mu_i \leq z^*$, the closest $z = \sum \mu_i x_i$, $x \in \mathsf{B}$, to z^* is given by $x_i^* = 1$ for each $i \in \mathsf{N}$. This x^* is the solution to (5.10). If $\sum \mu_i > z^*$, we must look for x_1^* and x_2^* in B which minimize the distances between $\sum \mu_i x_{ji}^*$ and z^*, where $\sum \mu_i x_{1i}^* \leq z^*$ and $\sum \mu_i x_{2i}^* \geq z^*$. Namely, we need to solve two subset sum problems written below, where z^* has been replaced by $\lfloor z^* \rfloor$ and $\lceil z^* \rceil$ because of the integrality of μ and x:

$$\begin{aligned} \max \quad & \sum_{i \in \mathsf{N}} \mu_i x_i \\ \text{s.t.} \quad & \sum \mu_i x_i \leq \lfloor z^* \rfloor \\ & x \text{ binary}, \end{aligned} \tag{5.14}$$

and

$$\begin{aligned} \min \quad & \sum_{i \in \mathsf{N}} \mu_i x_i \\ \text{s.t.} \quad & \sum \mu_i x_i \geq \lceil z^* \rceil \\ & x \text{ binary}. \end{aligned} \tag{5.15}$$

Denote by x_1^* and x_2^* solutions to (5.14) and (5.15). A solution to (5.10) is given by $x^* \in \{x_1^*, x_2^*\}$ such that $Z_{\mathcal{R}}(x^*) = \max(Z_{\mathcal{R}}(x_1^*), Z_{\mathcal{R}}(x_2^*))$.

Problems (5.14) and (5.15) are weakly polynomial, because they are particular cases of the knapsack problem. Then, $\lfloor z^* \rfloor$ and $\lceil z^* \rceil$ can be computed in $O\left(\log_2 \sum \mu_i\right)$, using a dichotomic search based on the sign of $\hat{Z}'$.

Fixed weights

We show next a result similar to Theorem 5.9 when w_i, $i \in \mathsf{N}$, are fixed and C is an arbitrary random variable.

Theorem 5.12. *Consider the problem*

$$\max_{x \in \mathsf{B}} \sum w_i x_i - K\mathbb{E}_C[\max(0, \sum w_i x_i - C(\omega))], \tag{5.16}$$

where C is a random variable. Problem (5.16) *is weakly $\mathcal{NP}$-hard and can be solved by a pseudo-polynomial algorithm in $O(n \sum w_i)$.*

Proof. The reduction from Proposition 5.2 holds when the variance of C is zero and K is large enough. Let Z be the objective function from (5.16). We must prove that Z has the same properties as $Z_{\mathcal{R}}$ from previous section so that the same argument as in the proof of Theorem 5.9 can be applied:

- Z depends only on $z = \sum w_i x_i$: $Z(x) = \hat{Z}(z)$.
- $\hat{Z} : [0, \sum w_i] \to \mathbb{R}$ is concave.
- We can compute $\lceil z^* \rceil$ and $\lfloor z^* \rfloor$ in $O(n \sum w_i)$.

It is straightforward to see that Z depends only on $z = \sum w_i x_i$, with $\hat{Z}(z) = z - \mathbb{E}_C[\max(0, z - C(\omega))]$.

Then, the concavity of Z (and therefore $\hat{Z}$) follows from Theorem 5.1. Using $\delta > 0$ small enough, we can use compute $\lceil z^* \rceil$ and $\lfloor z^* \rfloor$ by a dichotomic search based on the sign of $\hat{Z}(x) - \hat{Z}(x + \delta)$. □

5.3 Algorithms and computational experiments

In this section, we assume that random vector ξ is absolutely continuous, with density function f_ξ. One of the main difficulty of (**KSR**) is to evaluate the concave expectation term from (5.17):

$$\begin{aligned}\mathcal{Q}(x) &= -K\mathbb{E}_\xi\left[\max\left(0, \sum_{i=1}^n w_i(\omega)x_i - C(\omega)\right)\right] \\ &= -K\int_\Xi f_\xi(\xi)\max\left(0, \sum_{i=1}^n w_i x_i - C\right) dC dw_1 \ldots dw_n. \qquad (5.17)\end{aligned}$$

For an arbitrary continuous random vector ξ, the multivariate integral (5.17) is non-trivial and must be solved using efficient packages for numerical integration, see Prékopa (1995). To avoid this computational burden, we restrict ourselves to particular cases involving Gaussian and uniform distributions in the next subsections. Notice that for special distributions, such as the exponential, the recourse function has a closed form (Hansotia, 1977).

5.3.1 General case

This section describes a general algorithm to solve convex MINLPs and apply to (**KSR**) with Gaussian weights.

LP/NLP algorithm

In this Section, we briefly present an algorithm for solving linearly constrained convex MINLPs. Although (**KSR**) is unconstrained, we provide below a description for linearly constrained programs because the algorithm is used again in Chapter 6 for a network design problem featuring flow conservation constraints. Consider the following type of MINLP

$$(\mathbf{P}) \qquad \begin{array}{rl} \max & h(x) \\ \text{s.t.} & Ax \le b \\ & x \ge 0 \text{ and integer,} \end{array}$$

where h is assumed concave and differentiable. Problem (**P**) belongs to the class of mixed-integer non-linear programs, which have witnessed a tremendous attention in recent years. Efficient algorithms and solvers are now available to handle convex MINLP, see (Abhishek et al., 2010; Bonami et al., 2008, 2010). Moreover, results from Bonami et al. (2010) suggest that LP/NLP algorithms are particularly efficient to handle problems with linear constraints only, such as (**P**). Notice that these particular LP/NLP algorithms turn out to be branch-and-cut algorithms where cuts are nothing but linearizations of the objective function. The main points of our LP/NLP algorithm are detailed in what follows.

Because h is concave and differentiable, (**P**) is equivalent to

$$\begin{aligned} \max \quad & \gamma \\ \text{s.t.} \quad & h(\overline{x}) + \sum_{i=1}^{n} \frac{\partial h}{\partial x_i}(\overline{x})(x_i - \overline{x}_i) \geq \gamma \qquad \overline{x} \in \mathbb{R}^n_+ \\ & Ax \leq b \\ & x \geq 0 \text{ and integer,} \end{aligned} \tag{5.18}$$

which has an infinite number of constraints. The main idea of outer-approximation is that, for a given sensibility parameter $\epsilon > 0$, only a finite number of those constraints are required in a solution. For a given cut pool R, we define the upper bounding problem

$$(\mathbf{MP}) \qquad \begin{aligned} \max \quad & \gamma \\ \text{s.t.} \quad & h(\overline{x}) + \sum_{i=1}^{n} \frac{\partial h}{\partial x_i}(\overline{x})(x_i - \overline{x}_i) \geq \gamma \qquad \overline{x} \in \mathsf{R} \\ & Ax \leq b \\ & x \geq 0 \text{ and integer.} \end{aligned}$$

Our algorithm `lp/nlp` solves (**MP**) with the branch-and-cut described in Algorithm 4. This algorithm is similar to `bc-n`, from Chapter 3, with two differences. First we add cuts only at the root and at integer nodes, like in `bc-int`. Second, we differentiate h to (possibly) add a linearization instead of solving Linear Programs to add Benders cuts. T represents the branch-and-bound tree and solving a node $o' \in \mathsf{T}$ means solving the LP relaxation of (**MP**) augmented with branching constraints of o'. We show in Chapter 6 how to improve `lp/nlp` to take into account the (multi-commodity flow) structure of network design problems through Dantzig-Wolfe decomposition.

Computational results

We present next results of `lp/nlp` for solving (**KSR**) with Gaussian weights and a fixed capacity. As previously mentioned, more general distributions could be handled as long as numerical integration packages are available. Recall that when each w_i is a Gaussian $\mathcal{N}(\mu_i, \sigma_i)$ and C is a constant, (**KSR**) can be rewritten as

$$\max_{x \in B} \sum_{i \in N} p_i x_i - K \left\{ \sigma(x) f\left(\frac{C - \mu(x)}{\sigma(x)}\right) + (\mu(x) - C) G\left(\frac{C - \mu(x)}{\sigma(x)}\right) \right\}, \tag{5.19}$$

Algorithm 4: `lp/nlp`

```
begin                                                    /* Initialization */
    $\mathsf{T} = \{o\}$ where $o$ has no branching constraints;
    $UB = +\infty$;
while $\mathsf{T}$ is nonempty do
    select a node $o' \in \mathsf{T}$;
    $\mathsf{T} \leftarrow \mathsf{T} \backslash \{o'\}$;      /* withdraw node $o'$ from the tree */
    solve $o'$;
    let $(\overline{\gamma}, \overline{x})$ be an optimal solution;
    if $\overline{\gamma} < UB$ then
        if $\overline{x} \notin \mathbb{Z}^{|\mathsf{N}|}$ and $depth(o') \geq 1$ then
            branch, resulting in nodes $o^*$ and $o^{**}$;
            $\mathsf{T} \leftarrow \mathsf{T} \cup \{o^*, o^{**}\}$;      /* add children to the tree */
        else if $\overline{\gamma} \geq h(\overline{x}) + \epsilon$ then
            add $\overline{x}$ to $\mathsf{R}$;
            $\mathsf{T} \leftarrow \mathsf{T} \cup \{o'\}$;      /* put node $o'$ back in the tree */
        if $\overline{x} \in \mathbb{Z}^{|\mathsf{N}|}$ and $\overline{\gamma} < h(\overline{x}) + \epsilon$ then
            $UB \leftarrow \overline{\gamma}$;      /* define a new upper bound */
            $x^* \leftarrow \overline{x}$;      /* save current incumbent */
return $x^*$
```

where $\mu(x) = \sum_{i=1}^n \mu_i x_i$, $\sigma^2(x) = \sum_{i=1}^n \sigma_i^2 x_i^2$, f is the density function of $\mathcal{N}(0,1)$, $G = 1 - \Phi$ and Φ is the distribution function of $\mathcal{N}(0,1)$. Note that function G is read from a table. Algorithm 4 is implemented within CPLEX 11 (IBM-ILOG, 2009), with $\epsilon = 0.1$. Since the model does not contain explicitly all constraints, we must deactivate the dual presolve, setting *BooleanParam.PreInd* to false. Then, we implemented our (global) cut generation with a *LazyConstraintCallback*, preventing CPLEX from using the dynamic search. The algorithm has been coded in JAVA on a HP Compaq 6510b with a processor Intel Core 2 Duo of 2.40 GHz and 2 GB of RAM memory. We fix a time limit of 100 seconds per instance and the solution time has been set to 100 seconds for instances who could not be solved within this time limit or who exceeded the available memory.

We generated randomly different sets of instances, inspired by the instances from Martello et al. (1999). We consider two data ranges: $R = 1000$ and $R = 10000$. Then, parameters μ_i and p_i for each item $i \in \mathsf{N}$ are integers uniformly generated between 4 and R. Each variance σ_i is an integer uniformly generated between 1 and $\lfloor \mu/4 \rfloor$ for each item $i \in \mathsf{N}$, so that negative outcomes are negligible as explained next. We generated 100 instances for each value of parameters K, n and R, with a capacity $C = (h/101) \sum \mu_i$ for instance number h; all results take the average over the groups of 100 instances.

Note that Gaussian variables can take negative values which does make sense in real applications. Nevertheless, when the ratio σ/μ is small enough, the probability that this happens is negligible so that it does not affect sensibly the objective. For instance, the probability that $\mathcal{N}(\mu, \sigma^2)$ takes a value less than $\mu - 4\sigma$ is slightly less than 0.0001. Therefore we will assume $\mu/4\sigma \geq 1$ when generating our instances.

The following results show that the the penalty factor has little impact on the solution times, while a larger number of items makes the problem more difficult to solve. We study more specifically the time spent at the root node of Algorithm 4. The tables below report the total time in seconds (Time), the fraction of time spent at the root node (Initialization), the number of cuts generated at the root node and the number of cuts added deeper in the tree. We can see in Table 5.1 that the

K/R	Time		Initialization		InitCuts		AddCuts	
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
2	0.346	0.332	65%	65%	8.33	8.26	0.69	0.73
4	0.369	0.357	64%	64%	8.83	8.73	0.97	0.85
6	0.358	0.357	64%	65%	8.98	8.98	0.98	0.78
8	0.362	0.36	65%	66%	9.09	9.06	0.79	0.78
10	0.343	0.341	64%	63%	9.15	9.18	0.8	0.91
12	0.341	0.341	63%	62%	9.23	9.19	0.98	1.0
14	0.341	0.379	65%	65%	9.33	9.28	0.79	0.84
16	0.352	0.363	67%	66%	9.37	9.1	0.7	0.78
18	0.381	0.361	66%	64%	9.46	9.14	0.76	0.86
20	0.386	0.369	65%	66%	9.71	9.3	0.88	0.81

Table 5.1: Uncorrelated Instances, $n = 500$.

penalty factor K has little influence on `lp/nlp`. Therefore, we fix $K = 10$ in the other tests. Results from Table 5.2 show that we can easily solve problems up to 5000 variables, even though times are significantly larger than in the deterministic case. For example, uncorrelated instances with 5000 items are solved by Martello et al. (1999) on average in 0.01 seconds, whereas we need on average 19 seconds to solve such problems. Note that an important fraction of the time is spent in the generation of the cut pool at the root node, because most instances need to explore less than 100 nodes in their branch-and-cut trees. Pursuing our comparison with the deterministic knapsack, we wondered whether strongly correlated and Avis instances (Martello et al., 1999; Chvátal, 1980) are harder to solve than uncorrelated ones. Recall that strongly correlated instances are characterized by the relations $p_i = \mu_i + R/10$, $i \in \mathsf{N}$, while Avis instances are defined as follows: p_i is an integer uniformly generated between 1 and 1000, $\mu_i = n(n+1) + i$, and $C = n(n+1)\lfloor (n-1)/2 \rfloor + n(n-1)/2$. Results from Table 5.3 show that strongly correlated instances are roughly of the same difficulty as the uncorrelated ones, whereas strongly correlated ones are harder in the deterministic case. Avis instances are significantly harder to solve, see Table 5.4. Column (Unsolved) reports the number of unsolved instance within 100 seconds. While other problems were essentially solved a the root node, solving even small

n/R	Time		Initialization		InitCuts		AddCuts	
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
200	0.072	0.073	54%	49%	9.04	8.77	1.09	1.28
400	0.209	0.193	65%	62%	9.08	8.75	0.98	0.95
600	0.381	0.353	70%	72%	8.85	9.03	0.94	0.59
800	0.616	0.579	75%	77%	8.96	8.97	0.66	0.49
1000	0.896	0.876	78%	78%	8.73	8.92	0.61	0.61
2000	3.447	3.361	83%	84%	8.81	9.2	0.5	0.42
3000	7.329	7.5	86%	87%	8.85	9.22	0.34	0.28
4000	13.302	12.682	86%	87%	8.93	8.9	0.42	0.25
5000	20.405	17.462	86%	86%	9.02	7.8	0.4	0.28

Table 5.2: Uncorrelated Instances, $K = 10$.

Avis required to spend a large amount of time exploring branch-and-cut trees. In fact, unreported results show that thousands of nodes where required to solve Avis instances, while uncorrelated and strongly correlated instances were usually solved by exploring less than a hundred of nodes.

n/R	Time		Initialization		InitCuts		AddCuts	
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
200	0.122	0.149	29%	26%	8.46	8.66	2.27	2.22
400	0.353	0.374	33%	31%	8.64	8.6	1.86	1.94
600	0.927	0.718	25%	33%	8.24	8.42	1.81	1.51
800	0.842	1.064	47%	36%	8.23	8.16	1.17	1.33
1000	2.09	1.006	30%	57%	8.05	8.16	0.89	0.99
2000	3.155	2.727	76%	78%	7.91	7.85	0.59	0.4

Table 5.3: Strongly Correlated Instances, $K = 10$.

n	Time	Unsolved	Initialization	InitCuts	AddCuts
200	6.639	0	0.23%	11.34	1.79
400	29.379	6	0.05%	12.55	1.52

Table 5.4: Avis Instances, $K = 10$.

5.3.2 Pseudo-polynomial cases

In Section 5.2 we proved that special cases of (**KSR**) can be solved in pseudo-polynomial time. Since we provide constructive proofs, a natural question is to find out whether the pseudo-polynomial algorithms proposed in the proofs perform better than `lp/nlp`.

In all the following results, we fix a time limit of 100 seconds per instance and the solution time has been set to 100 seconds for instances who could not be solved within this time limit or who exceeded the available memory.

Fixed weights and uniformly distributed capacity

Lemma 5.5 shows how the solution to problem (5.6) can be obtained from the solutions to two knapsack problems and to the problem of maximizing the concave and quadratic function of binary variables (5.9). The computational results from Martello et al. (1999) show that the two knapsack problems can be solved in a very short amount of time, and we provide below results of `lp/nlp` applied to (5.9). Notice that other methods than `lp/nlp` can be used to solve (5.9). As mentioned in the proof of Lemma 5.8, (5.9) can be identified to a half-product that can be solved in pseudo-polynomial time. However, the algorithm from Badics and Boros performed much worse than `lp/nlp`, because very large numbers of states needed to be enumerated. Since the algorithm could hardly solve small instances with 200 variables, we do not report these computational experiments. Commercial MIP solvers are also able to handle (5.9). We provide below a numerical comparison of CPLEX 11 and `lp/nlp` described in Section 5.3.1. The instances are generated as follows.

K/R	`lp/nlp`		`cplex`				Time ratios	
	Time		Time		Unsolved			
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
2	0.019	0.02	0.019	0.038	0	1	0.7	0.91
4	0.018	0.02	0.036	0.943	0	0	0.78	0.82
6	0.02	0.021	0.246	5.718	0	0	0.77	1.27
8	0.019	0.023	1.311	9.913	0	1	1.05	0.59
10	0.02	0.022	2.598	11.894	0	1	1.15	0.91
12	0.02	0.022	3.233	14.764	0	1	1.11	0.94
14	0.021	0.022	3.299	17.98	0	0	1.09	3.93
16	0.021	0.023	3.329	21.403	1	0	1.29	6.15
18	0.022	0.025	4.662	25.744	0	2	1.14	1.19
20	0.021	0.024	5.256	28.096	0	2	1.23	1.39

Table 5.5: Comparison between `lp/nlp` and `cplex` when K increases, $n = 100$.

The parameters w_i and p_i for each item $i \in \{1, \dots, n\}$ are integers uniformly generated between 1 and R. For each data range R, each value of the penalty factor K and number of items n, we generate 100 instances, with $\mathbb{E}_\xi(C) = (h/101) \sum w_i$ for instance number h. Capacity C varies uniformly between 90% of $\mathbb{E}_\xi(C)$ and 110% of $\mathbb{E}_\xi(C)$. We report on Tables 5.5 and 5.6 the average solution time in seconds and the number of unsolved instances for `cplex` only because `lp/nlp` could solve all of them within the time limit. Ratios, given by ($Time$ `cplex`) / ($Time$ `lp/nlp`), are computed for each instance separately; we report the geometric average of the ratios, whereas we report the arithmetic average of solution times. We compute geometric average for ratios because of the following observation : if the ratio for one instance is 1/2 and the one for another instance is 2, their "average" should be equal to one, which is the case using the geometric average.

From Table 5.5, we see that the `lp/nlp` performance does not depend on the value of K, and that instances for the two range values are of the same difficulty. However `cplex` requires more time to solve instances with $R = 10^4$ than those with $R = 10^3$, which becomes even more significant when K increases. Even tough `cplex` takes on average more time than `lp/nlp`, the ratios close to one tell us that some instances are still solved faster by `cplex` than by `lp/nlp`. Table 5.6 studies the impact of

n/R	`lp/nlp` Time		`cplex` Time		`cplex` Unsolved		Time ratios	
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
200	0.029	0.032	0.08	0.269	0	0	1.93	1.77
400	0.052	0.057	0.484	0.239	0	0	5.33	4.14
600	0.076	0.081	1.878	0.638	1	0	9.51	7.75
800	0.097	0.109	2.083	1.447	0	0	14.83	13.31
1000	0.12	0.129	3.814	2.607	0	0	22.19	20.26
1200	0.144	0.166	4.176	4.064	2	1	28.52	28.29
1400	0.174	0.19	5.797	5.727	1	0	31.88	30.33
1600	0.203	0.222	6.982	7.984	1	0	34.8	36.09
1800	0.225	0.249	10.708	11.142	0	0	43.05	44.94
2000	0.264	0.281	12.313	14.309	2	0	48.28	50.86

Table 5.6: Comparison between `lp/nlp` and `cplex` when n increases, $K = 2$.

increasing the number n of items, hence variables in the model. It is clear from the values of the ratios that the `lp/nlp` handles better bigger instances than `cplex` does, the ratio average increases more or less linearly with the number of items. This may be explained by the following. `cplex` deals with $O(n^2)$ variables so that its solution time is very impacted by n. On the other hand, the number of variables in `lp/nlp` only increases linearly with n, because this algorithm deals implicitly with the non-linear objective. Then, because K is small enough, solution times required by `cplex` to solve instances with $R = 10^3$ are similar to those required to solve instances with $R = 10^4$. We stopped our test to 2000 variables because `cplex` required almost all memory to solve these instances.

Subset sum with gaussian weights

Our last group of tests studies subset sum instances with Gaussian weights, which satisfy the assumptions of Theorem 5.9. Therefore, μ_i are (integer) uniformly distributed between 1 and R and $\sigma_i = \sqrt{\lambda\mu_i}$, where λ must be chosen between 0 and 1. We have two algorithms at our disposal to solve these problems. As for general knapsack problem with Gaussian weights, we can use `lp/nlp` described in Section 5.3.1. Alternatively, using Theorem 5.9, we can solve these problems by the algorithm described in Algorithm 5. The later essentially solves two subset-sum problems using the method `decomp` from Pisinger (1999), available at `www.diku.dk/hjemmesider/ansatte/pisinger/`. Algorithm 5 has been coded in C on the same computer as the one used for `lp/nlp`. Table 5.7 studies the sensibility

Algorithm 5: `stoch-subsum`

```
1 compute $\lfloor z^* \rfloor$ and $\lceil z^* \rceil$ using a dichotomic search;
2 solve (5.14) and (5.15) with decomp from Pisinger (1999), yielding solutions $x_1^*$ and $x_2^*$;
3 if $Z_{\mathcal{R}}(x_1^*) > Z_{\mathcal{R}}(x_2^*)$ then $x^* := x_1^*$ else $x^* := x_2^*$;
  return $x^*$
```

to parameter λ for `lp/nlp`, results for `stoch-subsum` are not reported because all problems are solved within 0.001 seconds. As expected, Theorem 5.9 enables us to solve subset sum problems orders of magnitude faster than using `lp/nlp`, which is even more striking in Table 5.8 below.

λ/R	Time		Initialization		InitCuts		AddCuts	
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
1/64	1.783	1.979	7%	5%	11.14	7.03	2.11	1.42
1/16	1.625	1.852	8%	3%	10.9	11.19	1.58	1.63
1/4	1.609	1.891	6%	3%	10.42	10.97	1.45	1.96

Table 5.7: Subset Sum with $n = 500$ and $K = 10$.

To respect the condition $\mu/4\sigma \geq 1$, λ must satisfy $\lambda \leq \mu/16$. This becomes $\lambda \leq 1/16$ since μ is comprised between 1 and R. Therefore, we fix $\lambda = 1/16$ in our subsequent computational results. Table 5.8 compares `lp/nlp` and `stoch-subsum` for different values of n. Whenever `stoch-subsum` was able to solve an instance in less than 0.001 seconds, its solution time was set to 0.001 seconds. Ratios, given by (*Time* `lp/nlp`) / (*Time* `stoch-subsum`), are computed for each instance separately; we report the geometric average of the ratios, whereas we report the arithmetic average of solution times. Comparing Table 5.8 with Tables 5.2 and 5.3, we see that stochastic subset sum problems are significantly harder to solve than uncorrelated and correlated instances. This is due to the size of the branch-and-cut trees, hundreds of nodes being explored for the subset sum instances.

n/R	`lp/nlp`				`stoch-subsum`		Time ratios	
	Time		Unsolved		Time			
	10^3	10^4	10^3	10^4	10^3	10^4	10^3	10^4
200	0.199	0.244	0	0	0.001	0.001	170	198
400	0.917	0.991	0	0	0.001	0.001	636	736
600	2.542	2.726	0	0	0.002	0.002	1331	1364
800	5.306	5.568	0	0	0.002	0.002	2190	2096
1000	10.12	9.716	0	0	0.003	0.003	3422	3085
2000	84.768	79.023	31	26	0.005	0.005	12660	11475

Table 5.8: Comparison of `lp/nlp` and `stoch-subsum` on subset sum instances with $K = 10$ and $\lambda = 1/16$.

Similarly to the Avis knapsack instances, we can define Avis subset sum instances as follows: $p_i = \mu_i = n(n+1) + i$, and $C = n(n+1)\lfloor(n-1)/2\rfloor + n(n-1)/2$. To obtain groups of 100 "different" instances, we shuffle the order in which the items are read. Avis subset sum are extremely hard to solve already in the deterministic case, specialized algorithms are needed to solve large instances. Table 5.9 compares `lp/nlp` and `stoch-subsum` on these difficult instances. In fact, `lp/nlp` can not solve problem with more than 20 variables within the time limit of 100 seconds. This is due to the very large number of nodes explored. For $n = 10$, `lp/nlp` explores around 751 nodes on average, while exactly 705430 nodes are explored for each instance with $n = 20$. Although `stoch-subsum` requires more time than for other subset sum problems, it can still handle problems containing up to 600 variables within 100 seconds of CPU time.

n	`lp/nlp`	`stoch-subsum`	Time ratios
10	0.055	0.001	51.8
20	40.124	0.001	40119
100	–	0.055	–
200	–	1.003	–
300	–	5.22	–
400	–	16.6	–
500	–	41.6	–
600	–	88.1	–

Table 5.9: Comparison of `lp/nlp` and `stoch-subsum` on Avis subset sum instances with $K = 10$ and $\lambda = 1/16$.

6

Dantzig-Wolfe decomposition for MINLP applied to stochastic network design

6.1 Introduction

In Chapter 5, we saw how MINLP tools can be efficiently used to optimize the knapsack problem with simple recourse, as long as the expectation is easy to compute. Namely, we restricted ourselves to Gaussian random variables because their summation property avoids the burden of computing complicate multivariate integrals. In this chapter, we seek to extend these good computational results to a simple recourse network design problem. Formulating the network design problem as a convex MINLP yields a difficult problem, partly because of the large number of constraints and variables of the problem. As in Chapter 3, decomposition algorithms should be devised. In this chapter, we study another type of decomposition scheme, the Dantzig-Wolfe decomposition.

Many difficult Mixed-Integer Programs can be solved efficiently by Dantzig-Wolfe decomposition (abbreviated DWD in the sequel), followed by a branch-and-price algorithm, see Barnhart et al. (1998); Briant et al. (2008); Lübbecke and Desrosiers (2005), among others. Similarly to Benders decomposition, this reformulation may nicely split the problem into many smaller subproblems. For instance, generating new paths in the arcs-paths formulations for multi-commodity flow problems requires to solve a small shortest path problem for each commodity. Moreover, DWD provides a stronger bound than the linear relaxation of the problem when some complexity is transferred to the pricing problem. Nevertheless, since the number of variables in the reformulation is very large, one should rather generate them dynamically with a branch-and-price algorithm.

Independently, DWD has been successfully applied to linearly constrained problems with a pseudo-convex and differentiable objective, yielding the simplicial decomposition (Patriksson, 2009). Again, this decomposition replaces the possibly complicated constraints by the simple constraints defining the canonical simplex, but requires dynamic variable generation.

Up to our knowledge, no such decomposition has yet been applied to Mixed-Integer Non Linear Programs, and in particular convex MINLPs, although many efficient algorithms have been developed for convex MINLPs, see Bonami et al. (2009) for a review. Herein, we reformulate a convex linearly constrained MINLP using DWD. Then, we present a novel branch-and-cut-and-price algorithm based on the LP/NLP algorithm first introduced in Quesada and Grossman (1992) and implemented in in Chapter 5. The main difficulty of DWD lies in its implementation. Although it is nowadays very simple to code a Benders decomposition using cuts callbacks from most commercial (CPLEX, XPRESS, Gurobi) and non-commercial codes (SCIP, BCP, ...), see Chapter 3, commercial codes do not yet handle dynamic variable generation. Thus, implementing an efficient branch-and-cut-and-price algorithm is an important piece of work, beyond the scope of this thesis. Herein, we implement in JAVA (using the LP solver from CPLEX) a simplistic version of the algorithm and compare it with `lp/nlp` on the original formulation. We obtain encouraging results because our algorithm competes with CPLEX on five instances from SNDlib and always outperforms `lp/nlp`. Further work should improves our framework and extend it to more general convex MINLPs.

In the next section, we provide a literature review of network design under uncertainty and describe our stochastic network design problem with simple recourse. Then, our branch-and-cut-and-price algorithm is described in Section 6.3. Section 6.4 presents computational results.

6.2 Network design under demand uncertainty

Designing or extending a network is a costly task with duration ranging from a couple of months to many years. The parameters of the problem, such as demands and costs, vary along the time in a way which is usually impossible to predict exactly. At best, one can assess the evolution of these parameters with probability distributions (stochastic programming) or uncertainty polyhedrons (robust programming). To be relevant in practice, a model for network design or expansion planning must somehow take into account these uncertainties. In Chapters 3 and 4, we raised already the problem of edge failure, by means of models (**HOP**) and ($\mathbf{TEP_R - N1}$). In what follows, we are more interested in studying uncertainty related to the exact value of demands, d^q.

In this section, we review major works handling these uncertainties and present our model. We saw in Chapters 2, 3 and 4 that a network design problem can be modeled in various ways, depending on the practical application and on the level of simplification used. For the sake of simplicity, we restrict ourselves to model (**ND**) recalled in the next subsection, usually used to model telecommunications networks. The models presented in the following can be extended to more complex formulations, such as (**ML**),(**HOP**) or (**HOP**). H

6.2.1 Classical network design formulation

Let us recall the network design formulation introduced in Chapter 2. Given an undirected graph (V, E) and a set of commodities Q, with origin $s(q)$, destination $t(q)$, and nominal value d^q for every $q \in \mathsf{Q}$, the capacitated network design problem aims at installing the cheapest capacities on edges of G so that the resulting network shall be able to attend to each demand. Each edge $e \in \mathsf{E}$ between i and j can be used in both directions, so that we can introduce the set of arcs A making (V, A) a bi-directed graph. Integer variable y_e states how many batches of capacity C and cost c_e are installed on edge e, while fractional variable x^q_{ij} describes the amount of flow for commodity q through arc (i, j). The model reads as follows:

$$
\textbf{(ND)} \quad
\begin{aligned}
\min \quad & \sum_{e \in \mathsf{E}} c_e y_e && && (6.1)\\
\text{s.t.} \quad & \sum_{q \in \mathsf{Q}} \left(x^q_{ij} + x^q_{ji}\right) \le C y_e && e = ij \in \mathsf{E} && (6.2)\\
& \sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}} x^q_{ji} - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}} x^q_{ij} = \begin{cases} -d^q & \text{if } i = s(q) \\ d^q & \text{if } i = t(q) \\ 0 & \text{else} \end{cases} && i \in \mathsf{V},\ q \in \mathsf{Q} && (6.3)\\
& x^q_{ij} \ge 0 && && \\
& y_e \ge 0 \text{ and integer.} && && (6.4)
\end{aligned}
$$

Objective (6.1) minimizes design cost, constraints (6.2) ensure that capacities are not exceeded (note that flows in both directions of an edge e share the same capacity Cy_e), constraints (6.3) state that, for each commodity, the outgoing flow at every node of the graph must be equal to the ingoing flow apart from extremities $s(q)$ and $t(q)$. Finally, (6.4) ensures that capacities are installed by batches. We saw in Chapter 3 that (**ML**) can easily be derived from (**ND**) above, because the formulations of both problems have essentially the same structure. Hence, a natural question would be to address also models for (**ML**) under demand uncertainty. However, the deterministic version of (**ML**) is so difficult to solve to optimality that we did want to complicate even further the problem with the consideration of demand uncertainty. This could be an interesting topic for further research.

6.2.2 Dynamical routing

This general model considers that design decisions y must be taken before the demand is revealed, while routing decisions x can be fixed once we know the demand with precision. Said differently, x is an arbitrary functions of d.

Robust programming

Ben-Tal et al. (2004) introduce the *Adjustable Robust Counterpart* (ARC) which partitions decision variables into two sets. The first one contains variables that stay invariant regarding the value of uncertain parameters, while the variables in the second set can be fixed according to the specific values of these parameters, i.e., they are arbitrary functions of these parameters. Although ARC is untractable in general, Mattia (2010) applies that framework to (**ND**) leading to the problem of installing enough capacity y on the network such that for each d in the polyhedron $\mathcal{D}$, there exists a routing $x(d)$ feasible for y. Thus, x becomes a function of the demand, $x : \mathcal{D} \to \mathbb{R}_+$.

Stochastic programming

Given a set of scenarios Ω, the stochastic network design with routing recourse also partitions variables into two sets. As before, design variables must be fixed before demand is known, while routing can be devised independently for each scenario, $x : \Omega \to \mathbb{R}_+$. Moreover, we also introduce slack variables z allowing a portion of some of the demands to be unmet for scenarios with little probability weight. Namely, we replace (6.3) by

$$\sum_{j \in \mathsf{V}:(j,i)\in\mathsf{A}} x^q_{ji}(\omega) - \sum_{j \in \mathsf{V}:(i,j)\in\mathsf{A}} x^q_{ij}(\omega) = \begin{cases} -d^q(\omega) + z^q(\omega) & \text{if } i = s(q) \\ d^q(\omega) - z^q(\omega) & \text{if } i = t(q) \\ 0 & \text{else} \end{cases} \tag{6.5}$$
$$i \in \mathsf{V},\ q \in \mathsf{Q},\ \omega \in \Omega,$$

and add the penalty term $\sum_{q\in\mathsf{Q}} K^q \mathbb{E}[\max(0, z(\omega))]$ to objective (6.1). When capacity is allowed to be fractional (y is a continuous variable) Sen et al. (1994) devise a Monte-Carlo based decomposition algorithm for the problem and Lisser et al. (1999) implement a parallelized cutting-plane algorithm for the problem. With modular capacities, Andrade et al. (2005) implement a Benders decomposition and enhance the branching procedure. Riis and Andersen (2002) study a model including different routing schemes for each scenario as in (6.5) but without shortage variables z. The problem has therefore a structure close to (**ND**) so that they can extend well known classes of valid inequalities to the stochastic problem.

6.2.3 Oblivious routing

Dynamical routing suffers from two drawbacks. First, it is very hard computationally, unless efficient sampling algorithms are used, see Kleywegt et al. (2002) among others. Second, it may not describe accurately practical situations. In telecommunications networks for instance, an important problem is to prescribe a network able to cope with a set of different demands, corresponding to different periods of time. Often, it is not realistic to change completely the routing according to the exact demand value. For both reasons, it makes sense to apply some restrictions to function x. This idea of oblivious routing has already been applied with success in the context of robust programming, which we review quickly in the next subsection. In this thesis however, we shall stay within the framework of stochastic programming and introduce a new model for the problem.

Robust programming

In the context of robust programming, an important alternative to dynamical routing is the so-called static routing. Instead of having an arbitrary function $x : \mathcal{D} \to \mathbb{R}_+$, we introduce the notion of routing template

$$\sum_{j \in \mathsf{V}:(j,i)\in \mathsf{A}} f^q_{ji} - \sum_{j \in \mathsf{V}:(i,j)\in \mathsf{A}} f^q_{ij} = \begin{cases} -1 & \text{if } i = s(q) \\ 1 & \text{if } i = t(q) \\ 0 & \text{else} \end{cases} \qquad i \in \mathsf{V},\ q \in \mathsf{Q}, \tag{6.6}$$

and $x^q(d) = f^q d^q$. This model has been applied to a large panel of variations of (**ND**), see Altin et al. (2007, 2010); Mudchanatongsuk et al. (2008), among others. Ben-Ameur; Ouorou and Vial; Scutellà (2009), among others, propose more subtle routing schemes, in between static and dynamic routings.

Stochastic programming

Up to our knowledge, all stochastic programming approaches to (**ND**) with demand uncertainty consider dynamical recourse. We describe next a simple recourse model (**AN**) where x^q_{ij} denotes the maximal amount of flow for commodity q that can go on arc $(i,j) \in \mathsf{A}$, y_e represents the number of batches of capacity with size C installed on edge $e \in \mathsf{E}$, and z is the shortage recourse as in (6.5). Therefore, given the optimal solution $(\overline{y}, \overline{x}, \overline{z})$ to (**AN**), demand $d^q(\omega)$ is fully attended only if $\overline{z}^q(\omega) \le 0$. Let $\overline{d}^q = \max_{\omega\in\Omega}\{d^q(\omega) \text{ s.t. } \overline{z}^q(\omega) \le 0\}$. Hence, the actual flow on arc $(i,j) \in \mathsf{A}$ for commodity $q \in \mathsf{Q}$ is equal to

$$\overline{x}^q_{ij} \min\left(1, \frac{d^q(\omega)}{\overline{d}^q}\right).$$

The model reads as follows:

$$\begin{array}{lll}
 & \min & \displaystyle\sum_{e\in \mathsf{E}} c_e x_{ij} + \sum_{q \in \mathsf{Q}} K^q \mathbb{E}\left[\max(0, z^q(\omega))\right] \\
 & \text{s.t.} & \displaystyle\sum_{q\in \mathsf{Q}} \left(x^q_{ij} + x^q_{ji}\right) \le C y_e \qquad\qquad e = ij \in \mathsf{E} \\
(\mathbf{AN}) & & \displaystyle\sum_{j \in \mathsf{V}:(j,i)\in \mathsf{A}} x^q_{ji} - \sum_{j \in \mathsf{V}:(i,j)\in \mathsf{A}} x^q_{ij} = \begin{cases} -d^q(\omega) + z^q(\omega) & \text{if } i = s(q) \\ d^q(\omega) - z^q(\omega) & \text{if } i = t(q) \\ 0 & \text{else} \end{cases} \qquad (6.7) \\
 & & \qquad\qquad\qquad\qquad i \in \mathsf{V},\ q \in \mathsf{Q},\ \omega \in \Omega, \\
 & & x \ge 0 \\
 & & y \ge 0 \text{ and integer.}
\end{array}$$

The simple recourse structure of (**AN**) allows us to reformulate (**AN**) as a non linear problem, substituting $z^q(\omega)$ by

$$d^q(\omega) - \sum_{j \in \mathsf{V}: jt(q) \in \mathsf{A}} x^q_{jt(q)} + \sum_{j \in \mathsf{V}: t(q)j \in \mathsf{A}} x^q_{t(q)j},$$

as in Chapter 5, so that (6.7) becomes

$$\sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}} x^q_{ji} - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}} x^q_{ij} = 0 \qquad q \in \mathsf{Q},\ i \in \mathsf{V} \backslash \{s(q), t(q)\}. \tag{6.8}$$

Then, applying DWD to (**AN**) with (6.7) replaced by (6.8) and introducing continuous density probability functions $f^q : \mathbb{R}_+ \to \Xi^q$ for random variable d^q, we obtain our arcs-paths formulation:

$$(\mathbf{AP}) \quad \begin{array}{lll} \min & \displaystyle\sum_{e \in \mathsf{E}} c_e y_e + \sum_{q \in \mathsf{Q}} K^q \int_{\Xi^q} \max\left(0, \xi^q - \sum_{p \in \mathsf{P}^q} x^q_p\right) f^q(\xi^q) d\xi^q & (6.9) \\ \text{s.t.} & \displaystyle\sum_{q \in \mathsf{Q}} \sum_{p \in \mathsf{P}^q} \delta^p_e x^q_p \leq C_e y_e & e \in \mathsf{E} \\ & x \geq 0 & \\ & y \geq 0 \text{ and integer,} & \end{array}$$

where P^q is the set of all paths in G between $s(q)$ and $t(q)$, x^q_p the maximal flow on path p, and δ^p_e is equal to one if $e \in p$, 0 otherwise. Note that (6.8) defines a very special kind of polyhedron: a vector space. Therefore, its DWD only contains extreme rays (no vertices). This is clear from (**AP**) since no convexity constraints link path variables x^q_p.

Theorem 5.1 states that (6.9) is convex. Moreover, it is differentiable because each non-linear term of (6.9) can be rewritten

$$h^q(x) = K^q \int_{\Xi^q : \xi^q \geq \sum_{p \in \mathsf{P}^q} x^q_p} \left(\xi^q - \sum_{p \in \mathsf{P}^q} x^q_p\right) f^q(\xi^q) d\xi^q,$$

and $\sum_{p \in \mathsf{P}^q} x^q_p$ is differentiable. Remark that it may be difficult to compare fixed costs c and operating costs K so that we could replace the term $\sum_{e \in \mathsf{E}} c_e y_e$ in (6.9) by a budget constraint $\sum_{e \in \mathsf{E}} c_e y_e \leq B$, see Pióro and Medhi (2004) for examples of network design problems with budget constraints.

6.3 Algorithm

Problem (**AP**) has too many variables to be solved for real size networks, while only a few of them are required in the optimal solution. Hence, it would be interesting to generate paths only when needed. Once we have chosen a suitable MINLP framework, we must decide how to generate paths throughout the solution algorithm.

Herein, we decided to use the LP/NLP algorithm. Since (**AP**) has only linear constraints, we do not need to solve feasibility NLP so that the LP/NLP turns out to be a branch-and-cut algorithm, see `lp/nlp` from Chapter 5. Namely we define a master problem, see (**MP**) below, accumulating the linearizations of the objective function. Additional linearizations are generated throughout the branch-and-bound algorithm solving (**MP**), see Bonami et al. (2009). Although linearizations must be added at each integer node (to test whether we keep the associated incumbent), it is not obvious whether to add them at each fractional node. On one hand, it is important to add enough linearizations early in the tree to avoid exploration of too many infeasible nodes. On the other hand, adding too many unnecessary cuts would slow down the linear relaxation at each node, as discussed in Chapter 3 for Benders decomposition. Nevertheless, our algorithm `bp-lp/nlp` adds linearizations at every node of the enumeration tree.

Let R^q and and $\mathsf{P}^{q*} \subseteq \mathsf{P}^q$ be the sets of linearizations of h^q and the set of paths variables generated so far, respectively, for commodity q. We define below the master problem, (**MP**), for $|\mathsf{R}^q|$ linearizations of h^q and $|\mathsf{P}^{q*}|$ path variables for each commodity q:

$$(\mathbf{MP}) \quad \begin{array}{lll} \min & \sum_{e\in\mathsf{E}} c_e y_e + \sum_{q\in\mathsf{Q}} \gamma^q & \\ \text{s.t.} & h(\overline{x}) + \sum_{q\in\mathsf{Q}} \sum_{p\in\mathsf{P}^{q*}} \frac{\partial h}{\partial x_p^q}(\overline{x},\overline{y})(x_p^q - \overline{x}_p^q) \le \gamma^q & q \in \mathsf{Q},\, \overline{x} \in \mathsf{R}^q \quad (6.10) \\ & \sum_{q\in\mathsf{Q}} \sum_{p\in\mathsf{P}^{q*}} \delta_e^p x_p^q \le C_e y_e & e \in \mathsf{E} \quad (6.11) \\ & \gamma,\, x \ge 0 & \\ & y \ge 0 \text{ and integer,} & \end{array}$$

and define $\mathsf{P}^* = \cup_{q\in\mathsf{Q}} \mathsf{P}^{q*}$. Note that each h^q satisfies the following property:

$$\frac{\partial h^q}{\partial x_p^q} = \frac{\partial h^q}{\partial x_{p'}^q} \text{ for any } p, p' \in \mathsf{P}^q, \tag{6.12}$$

so that we denote (6.12) by $\frac{dh^q}{dx}$ in the following. Property (6.12) implies that the reduced cost is easy to compute for any path $p \in \mathsf{P}$.

Lemma 6.1. *Let* $(\overline{x}, \overline{y}, \overline{\gamma})$ *be an optimal solution to* (**MP**)*,* $\overline{u}$ *and* $\overline{v}$ *be optimal multipliers associated with constraints* (6.10) *and* (6.11)*, respectively, and* $p \in \mathsf{P}^q$ *for some* $q \in \mathsf{Q}$*. The reduced cost of* x_p^q *is equal to:*

$$\overline{c}_p^q = -\sum_{\overline{x}\in\mathsf{R}^q} \frac{dh^q}{dx}(\overline{x})\overline{u} + \sum_{e\in E} \delta_e^p \overline{v}_e. \tag{6.13}$$

Reduced costs tell us which paths in $\mathsf{P}\backslash\mathsf{P}^*$ may improve the objective of (**MP**), and should thus be added to P^*. What can we say about these paths concerning the objective of (**AP**)? Since (**AP**) is non-linear, we cannot, in general, compute reduced costs to know which (path) variables should be considered. Nevertheless, since for each $q \in \mathsf{Q}$, the piece-wise linear function defined by (6.10) is always smaller than or equal to h^q, we can use $\overline{c}_p^q$ to know whether new paths can improve significantly the objective of (**AP**), that is, improve it by more than a given $\epsilon > 0$. Let $h(\mathsf{P}^*)$ and $h(\mathsf{P}^*, \mathsf{R})$ denote the optimal solutions of (**AP**) restricted to paths in P^* and (**MP**) restricted to paths in P^* and with linearizations in $\mathsf{R} = \cup_{q \in \mathsf{Q}} \mathsf{R}^q$, respectively.

Lemma 6.2. *Given P^*, let $\mathsf{R} = \cup_{q\in\mathsf{Q}}\mathsf{R}^q$ be such that $h(\mathsf{P}^*, \mathsf{R}) \geq h(\mathsf{P}^*) - \epsilon$, and consider some path $p \in \mathsf{P}^q \backslash \mathsf{P}^{q*}$. If $\overline{c}_p^q \geq 0$, then $h(\mathsf{P}^* \cup \{p\}) \geq h(\mathsf{P}^*) - \epsilon$.*

Proof. Because $\overline{c}_p^q \geq 0$, path p does not improve the objective of (**MP**), that is $h(\mathsf{P}^*, \mathsf{R}) = h(\mathsf{P}^* \cup \{p\}, \mathsf{R})$. Therefore,

$$\begin{aligned} h(\mathsf{P}^* \cup \{p\}) &\geq h(\mathsf{P}^* \cup \{p\}, \mathsf{R}) \\ &= h(\mathsf{P}^*, \mathsf{R}) \\ &\geq h(\mathsf{P}^*) - \epsilon. \end{aligned}$$

□

As in Chapter 5 we linearize the non-linear objective only up to some $\epsilon > 0$. Lemma (6.2) implies that only a path p with strictly negative reduced cost can improve the objective of (**AP**) by more than ϵ. Thus, `bp-lp/nlp` below looks for a minimum to (**AP**) up to $\epsilon > 0$, so that paths with positive reduced cost can be neglected. Note that the term $\sum_{\overline{x} \in \mathsf{R}^q} \frac{dh^q}{dx}(\overline{x})\overline{u}$ depends only on commodity $q \in \mathsf{Q}$, not on path p. Thus, the pricing problem turns out to be a shortest path problem for each commodity, with edge costs $\overline{v}_e$. This is a well known problem polynomially solvable that we solve by linear programming.

Our algorithm is described on Algorithm 6. It is an extension of `lp/nlp` from Chapter 5 handling columns generation.

Remark 6.3. *Algorithm 6 is an "easy" branch-and-cut-and-price algorithm in the sense that we do not branch on the variables that are generated dynamically: we branch on the y variables and generate the x variables. To extend Algorithm 6 to problems with unsplittable flows, see* (**UND**) *in Chapter 7, one should use a more sophisticated branching procedure such as (Barnhart et al., 2000).*

6.4 Computational experiments

This section presents our computational experiments. First, we introduce our instances and the details of our implementation.

6.4.1 Test sets and implementation details

In what follows, we provide more details about our set of instances and the parameters used.

Algorithm 6: `bp-lp/nlp`

```
begin                                                    /* Initialization */
    $\mathsf{T} = \{o\}$ where $o$ has no branching constraints;
    $UB = +\infty$;
    cut $=$ true;
    var $=$ true;
while $\mathsf{T}$ is nonempty do
    select a node $o' \in \mathsf{T}$;
    $\mathsf{T} \leftarrow \mathsf{T} \setminus \{o'\}$;
    while cut $=$ true or var $=$ true do
        solve $o'$;
        let $(\overline{x}, \overline{y}, \overline{\gamma})$ be an optimal solution;
        let $(\overline{u}, \overline{v})$ be optimal dual multipliers;
        cut $=$ false, var $=$ false;
        if $c^t\overline{y} + \overline{\gamma} < UB$ then
            foreach $q \in \mathsf{Q}$ do
                if $h^q(\overline{x}) \geq \overline{\gamma}^q + \epsilon$ then
                    add $\overline{x}$ to $\mathsf{R}^q$;
                    cut $=$ true;
            if cut $=$ false then
                foreach $q \in \mathsf{Q}$ do
                    let $p$ be the shortest paths between $s(q)$ and $t(q)$ according to
                    costs $\overline{v}$;
                    if $\overline{c}^q_p < 0$ then
                        add $p$ to $\mathsf{P}^q$;
                        var $=$ true;
    if $\overline{y} \in \mathbb{Z}^{|\mathsf{E}|}$ then define a new upper bound $UB := c^t\overline{y} + \sum_{q\in\mathsf{Q}} \overline{\gamma}^q$ and save
    current incumbent;
    else
        branch, resulting in nodes $o^*$ and $o^{**}$;
        $\mathsf{T} \leftarrow \mathsf{T} \cup \{o^*, o^{**}\}$;
```

Instances

We detail results obtained on two sets of instances:

- Randomly generated graph with 50 nodes and 100 edges, with four batches of capacity allowed on each edge and $C = 100$. The number of end-to-end commodities varies between 10 and 20. Their demands follow Gaussian distributions with means uniformly distributed between 1 and 40, and each variance is uniformly distributed between 0 and the mean divided by four. Costs c are based on euclidean distances between nodes.

- SNDlib networks with details reminded in Table 6.1. Demands follow Gaussian distributions with means equal to the nominal values of the demands in SNDlib, and each variance is uniformly distributed between 0 and the mean divided by four.

Name	|V|	|E|	|Q|
dfn-gwin	11	47	110
newyork	16	49	240
polska	12	18	17
atlanta	15	22	55
nobel-ger	17	26	121

Table 6.1: Instances description.

Implementation

The following algorithms are coded in JAVA on a HP Compaq 6510b with a processor Intel Core 2 Duo of 2.40 GHz and 2 GB of RAM memory.

- `cplex`: Formulation (**AN**) solved by an `lp/nlp` with the branch-and-cut framework from CPLEX 12.1; cuts (6.10) are implemented through the *LazyConstraintCallback*.

- `lp/nlp`: Formulation (**AN**) solved by an `lp/nlp` fully implemented in JAVA using CPLEX 12.1 as the LP solver.

- `bp-lp/nlp`: Formulation (**AP**) solved by `bp-lp/nlp` fully implemented in JAVA using CPLEX 12.1 as the LP solver. We check for violated cut and missing path at every node of the tree.

6.4.2 Results

$\|Q\|$	Total time			(**MP**)		(**MP**) and price	time ratio
	`cplex`	`lp/nlp`	`bp-lp/nlp`	`lp/nlp`	`bp-lp/nlp`	`bp-lp/nlp`	`lp/nlp` / `bp-lp/nlp`
10	4.5	3.1	2.7	2.3	0.93	1.56	1.1
12	13.7	24.6	15.8	21.2	9	11.4	1.6
14	64	150	82	134	55	64.5	1.8
16	43	171	101	155	69	81	1.7
18	100	581	303	523	205	238	1.9
20	100	1602	742	1455	510	585	2.2

Table 6.2: CPU times on randomly generated instances.

Detailed results for random instances are given in Table 6.2, and total times are also depicted in Figure 6.1. Columns “Total time” present the total amount of CPU time required by each algorithm to solve the problem, columns “(**MP**)” provide only the time spent for solving the bounding problem with CPLEX, and column “(**MP**) and price” sums the times for solving the bounding problem and for pricing out new variables. Finally, column “time ratio” provides the ratios between total times of `lp/nlp` and `bp-lp/nlp`.

Our main objective here is show the improvement obtained by `bp-lp/nlp` when compared to `lp/nlp`, because both of them have been implemented in a naive fashion. The comparison with `cplex` is only made out of curiosity because the latter enjoys from a wide range of clever engineering improvements. In this sense, Table 6.2 and Figure 6.1 show that our branch-and-cut-and-price algorithm is a clear success since the total time ratio increases with the size of the problem.

Then, we depict in Figure 6.2 total times of the three algorithms applied to Sndlib networks from Table 6.1. The chart on left provides the absolute total times in seconds, while the chart of right compares these times with the time required by `lp/nlp`. Again, `bp-lp/nlp` clearly outperforms `lp/nlp`. Moreover, it also competes with `cplex`, which is very encouraging.

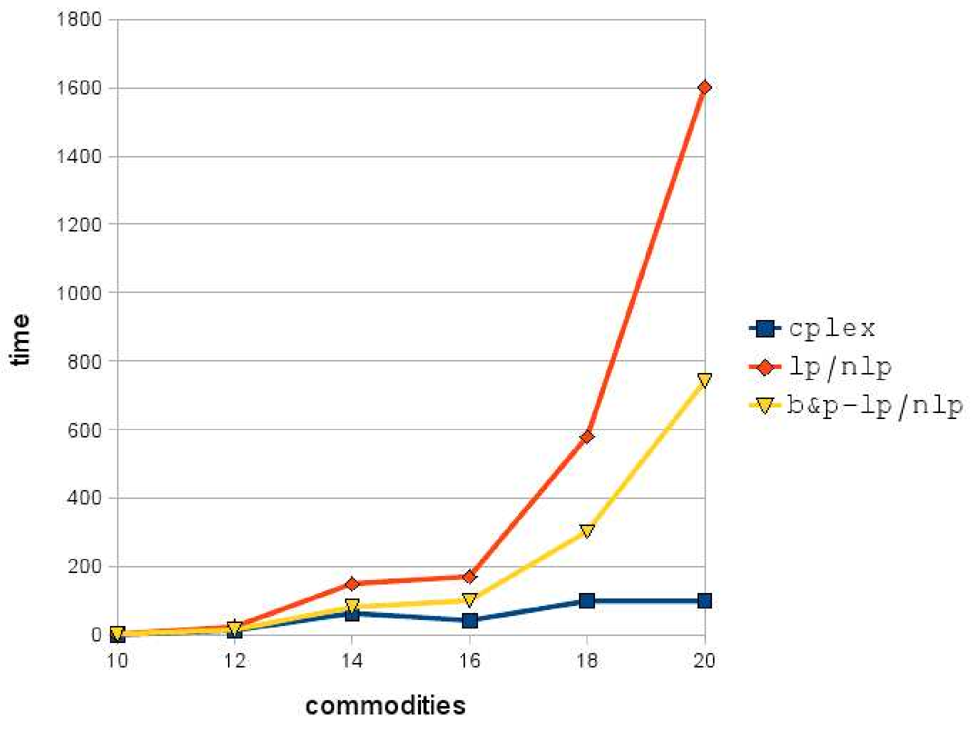


Figure 6.1: Solution times for randomly generated instances.

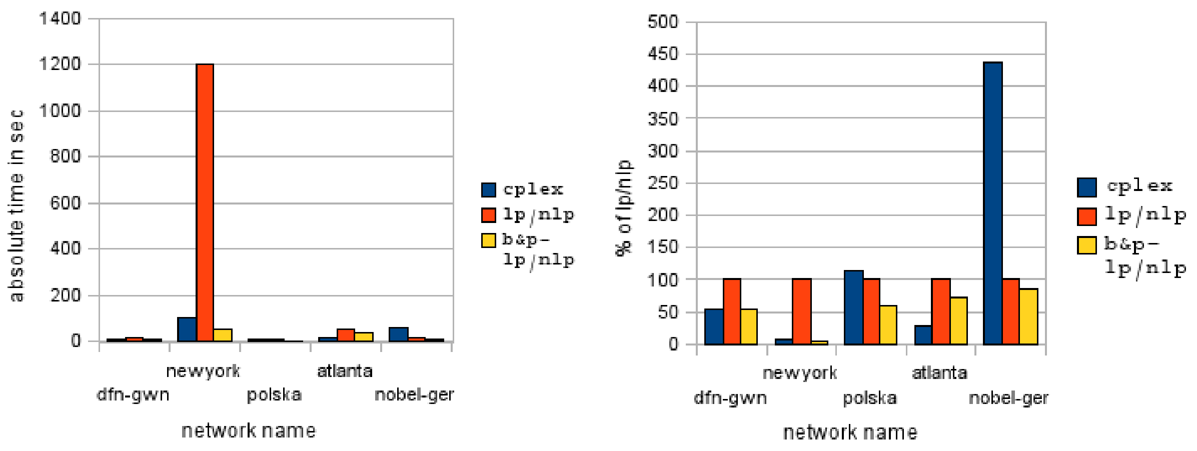


Figure 6.2: Solution times for SNDlib instances.

7

Easy distributions for combinatorial optimization problems with probabilistic constraints

7.1 Introduction

Many combinatorial optimization models address problems with parameters which are impossible to predict exactly. Therefore, it is often more accurate to model these parameters with random variables. This modifies the structure of the optimization problems, depending on the times at which decisions are taken and parameters are revealed. Chapters 5 and 6 presented two-stages models: some variables are to be fixed now while recourse variables are fixed after uncertain parameters are known exactly. A recourse model has also been mentionned in the last section of Chapter 4. In this chapter we study probabilistic constraints: all decisions must be taken here and now, such that the constraints of the model shall be satisfied with a certain probability. In other words, we aim at maximizing some objective for a given feasibility tolerance.

Stochastic programs with linear probabilistic constraints are in general non-convex non-linear optimization problems (Henrion and Strugarek, 2008). If furthermore some variables are integer, they become non-convex Mixed Integer Non Linear Problems (Grossmann, 2002). Although probabilistic constraints have been widely studied for many years, see Henrion (2004); Prékopa (2003); Shapiro et al. (2009) and the references therein, papers on problems with integer variables are not very numerous. Among them, problems featuring joint probabilistic constraints with random right hand side have been studied by Beraldi and Ruszczynski (2002b,a, 2005) who propose exact and heuristic branch-and-bound algorithms, Dentcheva et al. (2002) who study formulations and bounding procedures, Lejeune and Ruszczynski (2007) who develop a column-generation based algorithm for a supply chain management problem, and Saxena et al. (2009) who introduce the concepts of p-inefficiency and provide extensive computational results for the probabilistic set-covering problem studied by Beraldi and Ruszczynski (2002a). All these works handle probabilistic constraints through the concept of p-efficient points introduced by Prékopa (1990), apart from Saxena et al. (2009) who uses p-inefficient points instead.

Herein, we consider problems where uncertainty affects both sides of the constraints. A branch-and-bound algorithm and heuristics for such problems have been proposed by Beraldi and Bruni (2009, 2010), and Klopfenstein (2010) studies valid inequalities for the problem with individual probabilistic constraints with uncertainty in both sides.

In what follows, we are particularly interested by the case of individual probabilistic constraints while the random variables follow particular continuous distributions, among which normal distributions. Previous results in this direction assume that all random variables are normally distributed. In that case, the probabilistic constraints can be rewritten as quadratic constraints (Kataoka, 1963; Prékopa, 1995; van de Panne and Popp, 1963), convex under some assumption on the confidence level (Parikh, 1968). If all variables are binary, the constraints can be further linearized using classical techniques (Hansen and Meyer, 2009). Further work extends the classical Gaussian framework to the more general class of radial distributions (Calafiore and Ghaoui, 2006). The authors show how a probabilistic constraint can be written as a second-order cone convex constraint. The latter constraint can be linearized as well when working with binary variables.

In this chapter, we always assume that coefficients are independent continuous random variables. We show that an individual linear probabilistic constraint with binary variables is equivalent to a linear constraint when all coefficients are distributed according to either $\mathcal{N}(\mu_i, \lambda\mu_i)$, for some $\lambda > 0$ and $\mu_i > 0$, or $\Gamma(k_i, \theta)$ for some $\theta > 0$ and $k_i > 0$. The constraint can also be linearized when the coefficients are independent and identically distributed, if they are either positive or strictly stable random variables. As a result, we obtain that certain types of chance-constrained knapsack problems are as easy to solve as their deterministic counterpart.

The next section motivates the study of these constraints from a network design problem involving unsplittable multi-commodity flows. Then, in Section 7.3 we study the case of identically distributed random variables, while in Section 7.4 we study Gaussian and gamma random variables. Finally, Section 7.5 applies the results from Section 7.4 to the chance-constrained knapsack problem, and discusses how these results can be applied to a unsplittable multi-commodity flow problem arising in telecommunications networks.

7.2 Studied constraints

In this section we introduce formally the constraint studied in the chapter.

7.2.1 Unsplittable multi-commodity flow

In Chapters 3 and 4, we described three different network design problems motivated by real applications in telecommunications and power transmission networks. Among them, only (**HOP**) features integer routing variables forcing the flows to be sent along only one path for each commodity. Actually, (**HOP**) uses more than one path by duplicating the flows in order to respect survivability requirements. These integer variables yield the so-called "unsplittable" (sometimes called "non-bifurcated") routing schemes, required to model in a realistic way many telecommunications technologies. Considering these constraints within the standard formulation (**ND**) for capacitated network design, we obtain the following model

$$
(\mathbf{UND}) \quad \begin{aligned}
\min \quad & \sum_{e \in \mathsf{E}} c_e y_e \\
\text{s.t.} \quad & \sum_{q \in \mathsf{Q}} d^q \left(x_{ij}^q + x_{ji}^q \right) \le C y_e && e = ij \in \mathsf{E} && (7.1) \\
& \sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}} x_{ji}^q - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}} x_{ij}^q = \begin{cases} -1 & \text{if } i = s(q) \\ 1 & \text{if } i = t(q) \\ 0 & \text{else} \end{cases} && i \in \mathsf{V},\ q \in \mathsf{Q} && (7.2) \\
& x \text{ binary} && && (7.3) \\
& y \ge 0 \text{ and integer.}
\end{aligned}
$$

As with (**ND**), we are given an undirected graph $\mathsf{G} = (\mathsf{V}, \mathsf{E})$, where A denotes the set of directed arcs obtained by considering each $e \in \mathsf{E}$ in both directions. Variable y_e describes the capacity installed on edge $e \in \mathsf{E}$ and variable x_{ij}^q describes the flow for commodity $q \in \mathsf{Q}$ on arc $(i, j) \in \mathsf{A}$. The main difference between (**ND**) and (**UND**) is the use of binary routing variables (7.3) while (**ND**) is allowed to use fractional flow variables. Note that this required to move demand coefficient d^q from flows conservations constraints (7.2) to capacity constraints (7.1).

We already experienced one of the difficulty of (7.3), namely, the impossibility of projecting out x variables through Benders decomposition, unless the problem satisfies some additional property (see Proposition 3.1 from Chapter 3). The other difficulty of (7.3) is the substitution of $|\mathsf{Q}| \times |\mathsf{A}|$ continuous variables by binary ones, making the branch-and-bound tree very large unless clever cutting planes and/or branching schemes are devised.

In this chapter, we are interested in considering probabilistic constraints to handle demand uncertainty. Namely, d^q becomes a random vector (which we still note d^q to ease notations) so that each of the constraint in (7.1) is replaced by

$$
P\left(\sum_{q \in \mathsf{Q}} d^q \left(x_{ij}^q + x_{ji}^q \right) \le C y_e \right) \ge p, \tag{7.4}
$$

for p close to 1. With (7.4), (**UND**) can be considered a stochastic model that uses oblivious routing, see Chapter 6. For arbitrary random variables d^q, (7.4) is in general non-linear and non-convex, making the problem a hard MINLP. However, we explain in Sections 7.3 and 7.4 how to linearize (7.4) under strong assumptions about distributions of d^q. This simplification uses extensively the fact that x is a binary vector, so that (7.3) makes the problem with (7.4) somewhat simpler than if the routing were fractional. This contrasts with the deterministic version of (**UND**) for which (7.3) increases considerably the complexity of the problem.

7.2.2 General constraint

In the following we study mainly the following type of probabilistic constraints,

$$\mathcal{C}_1(x) = P\left(\sum_{i=1}^{n} w_i x_i \leq C_0\right) \geq p, \tag{7.5}$$

though our results extend easily to

$$\mathcal{C}_2(x) = P\left(\sum_{i=1}^{n} w_i x_i \leq C_1 y_1 + C_0\right) \geq p, \tag{7.6}$$

and

$$\begin{aligned} \mathcal{C}_3(x) \;=\; P\left(\sum_{i=1}^{n} w_i x_i \leq \sum_{j=1}^{m} C_j y_j + C_0\right) &\;\geq\; p \\ \sum_{j=1}^{m} y_j &\;\leq\; 1, \end{aligned} \tag{7.7}$$

where $p \in (0,1)$, w_i are independent random variables, and C_j, $0 \leq j \leq m$, are fixed coefficients. In addition, we always consider that $x_i, y_j \in \{0,1\}$, for $i \in \mathsf{N} = \{1,\dots,n\}$ and $1 \leq j \leq m$. The first constraint (7.5) is the so-called knapsack constraint, studied in Chapter 5, which plays an important role in capacitated problems such as unsplittable multicommodity flow and generalized assignment problems. The second constraint (7.6) appears when the choice of the capacitated facilities to be built is part of the decision: C_0 denotes the initial capacity and C_1 the capacity provided by the facility. Typical examples are network design problems and facility location problems. Finally, in many technical problems we must choose at most one out of a set of different facilities, for instance, different capacities for a new link to install in a telecommunication network. This is represented by (7.7). For example, (7.4) can be rewritten as (7.7), setting $w = d$, $C_0 = 0$ and $C_j = jC$, so that $y_j = 1$ if $y_e = j$, 0 otherwise.

In the sequel, we say that two constraints $C_1(x) \geq 0$ and $C_2(x) \geq 0$ are equivalent, denoted by $C_1(x) \geq 0 \Leftrightarrow C_2(x) \geq 0$, if the sets $\{x \in \{0,1\}^n \text{ s.t. } C_1(x) \geq 0\}$ and $\{x \in \{0,1\}^n \text{ s.t. } C_2(x) \geq 0\}$ are equal. Moreover, the summation $\sum$ refers to the sum over set $\mathsf{N} = \{1,\dots,n\}$ unless stated otherwise.

7.3 Identically distributed variables

We first consider (7.5) for the simple example where w_i are positive random variables identically distributed. Since w_i are positive, we see that

$$P\left(\sum_{i=1}^{m} w_i \le C_0\right) \le P\left(\sum_{i=1}^{m-1} w_i \le C_0\right). \tag{7.8}$$

Thus, the number of x_i that can be equal to 1 can certainly not exceed

$$N(C_0) = \max_{1\le l\le n}\left\{l \text{ s.t. } P\left(\sum_{i=1}^{l} w_i \le C_0\right) \ge p\right\}. \tag{7.9}$$

Conversely, if some binary vector x satisfies $\sum x_i \le N(C_0)$, then certainly x satisfies (7.5) because w_i are identically distributed. Then, considering (7.6), the previous reasoning holds with $N(C_0)$ for $y_1 = 0$, and with $N(C_0 + C_1)$ for $y_1 = 1$. Finally, this reasoning extends to the pair of constraints (7.7), since at most one of the y_j can be equal to 1. We just proved the following:

Proposition 7.1. *Consider n independent identically distributed positive random variables w_i, $i \in \mathsf{N}$. Then, for $x_i, y_j \in \{0,1\}$, $i \in \mathsf{N}$ and $1 \le j \le m$, the following constraints are equivalent:*

1. $\mathcal{C}_1(x) \ge p \Leftrightarrow \sum x_i \le N(C_0)$
2. $\mathcal{C}_2(x) \ge p \Leftrightarrow \sum x_i \le (N(C_0 + C_1) - N(C_0))y_1 + N(C_0)$
3. *If furthermore,* $\sum_{j=1}^{m} y_j \le 1$, *then*

$$\mathcal{C}_3(x) \ge p \Leftrightarrow \sum_{i=1}^{n} x_i \le \sum_{j=1}^{m}(N(C_0 + C_j) - N(C_0))y_j + N(C_0)$$

with $N(r)$ defined in (7.9) *for any real r.*

In the following, we focus on results of type *1.* since *2.* and *3.* can be deduced from *1.* by the above arguments. Hence, we denote C_0 by C in the sequel to simplify notations.

Remark that computing the value of $N(C)$ requires, in general, the solution of a multivariate integral that must be solved using efficient packages for numerical integration, see Prékopa (1995). For some distributions, this computational burden can be avoided. For instance, if all w_i are uniformly distributed between 0 and 1, their sum is distributed according to (Grinstead and Snell, 1997)

$$f(z) = \frac{1}{n!}\sum_{k=0}^{n+1}(-1)^k \binom{n+1}{k} \left[(z-k)_+\right]^n.$$

The uniform distributions are not the only distributions which sum up nicely. Stable distributions satisfy interesting summation properties too. Recall that if w_i are n independent copies of a stable random variable w, then for any constants x_i the random variable $\sum_{i=1}^n x_i w_i$ has the same distribution as $v_n\, w + w_n$ with some constants $v_n = n^{1/\alpha}$ for some $\alpha \in (0, 2]$, and w_n. Moreover, w is said strictly stable if $w_n = 0$ in the relation above. For instance, the Levy distribution, with density function equal to $f(z; c) = \sqrt{\frac{c}{2\pi}} \frac{e^{-c/2x}}{x^{3/2}}$ for $z \geq 0$, is positive (satisfying the hypothesis of Proposition 7.1) and stable so that sums of such distributions are easy to compute. We refer to Nolan (2010) for a good introduction to stable distributions.

In general, the support of stable distributions intersects negative reals. For instance, normal and Cauchy distributions always have negative tails. We show next that property (7.8) still holds for strictly stable distributions. For w_i strictly stable i.i.d. random variables, we have by definition that

$$\sum_{i=1}^{n} w_i \sim n^{1/\alpha} w_1 \qquad \alpha \in (0, 2],$$

so that

$$P\left(\sum_{i=1}^{n} w_i \leq C\right) = P(n^{1/\alpha} w_1 \leq C) = P(w_1 \leq C n^{-1/\alpha}).$$

If $C \geq 0$, the function $n \mapsto C n^{-1/\alpha}$ is non increasing, implying (7.8). We obtain the following:

Proposition 7.2. *Consider n independent identically distributed strictly stable random variables w_i, $i \in \mathsf{N}$, and $C \geq 0$. Then, if $x_i \in \{0, 1\}$ for each $i \in \mathsf{N}$, the following constraints are equivalent:*

$$\mathcal{C}_1(x) \geq p \Leftrightarrow \sum x_i \leq N(C),$$

with $N(C)$ defined in (7.9).

An example of strictly stable distribution with $\alpha = 1$ is the Cauchy distribution, with density function $f(z; z_0, \gamma) = \frac{1}{\pi}\left(\frac{\gamma}{(z - z_0)^2 + \gamma^2}\right)$ for some location parameter $z_0 \in \mathbb{R}$ and scale parameter $\gamma > 0$.

7.4 Non identically distributed variables

A well known stable distribution is the Gaussian distribution. In fact, for Gaussian and gamma random variables we are able to derive stronger results, allowing for the random variables to be distributed differently, as long as some regularity condition holds. Consider independent Gaussian random variables, $w_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, $i \in \mathsf{N}$. Then, $\mathcal{C}_1(x) \geq p$ can be rewritten (Prékopa, 1995)

$$\sum \mu_i x_i + \Phi^{-1}(p) \sqrt{\sum \sigma_i^2 x_i^2} \leq C, \tag{7.10}$$

where Φ is the distribution function of the standard normal distribution $\mathcal{N}(0,1)$. When $x \in \{0,1\}^n$, (7.10) can be linearized introducing additional continuous variables (Hansen and Meyer, 2009). However, these linearizations contain more variables and provide looser bounds than the direct linearization from Proposition 7.3 below.

Proposition 7.3. *Consider n random variables $w_i \sim \mathcal{N}(\mu_i, \lambda\mu_i)$, $i \in \mathsf{N}$, for $\lambda > 0$ and $\mu_i > 0$. Then, if $x_i \in \{0,1\}$ for each $i \in \mathsf{N}$, the following constraints are equivalent:*

$$\mathcal{C}_1(x) \geq p \Leftrightarrow \sum \mu_i x_i \leq \mu^*, \tag{7.11}$$

where μ^* is the unique root of the equation $C - \mu = \Phi^{-1}(p)\sqrt{\lambda\mu}$.

Proof. Recall that if $w_1, \ldots, w_n$ are independent Gaussian with mean μ_i and variance σ_i^2, and x_i are real numbers, then $w := \sum_{i=1}^n x_i w_i \sim \mathcal{N}(\mu(x), \sigma^2(x))$, with $\mu(x) = \sum x_i \mu_i$ and $\sigma^2(x) = \sum x_i^2 \sigma_i^2$. Thus, because $x_i \in \{0,1\}$ and $\sigma_i^2 = \lambda\mu_i$ for each $i \in \mathsf{N}$, we have $\sigma^2(x) = \lambda\mu(x)$. Then,

$$P\left(\sum w_i x_i \leq C\right) = P\left(\mathcal{N}(0,1) \leq \frac{C - \mu(x)}{\sqrt{\lambda\mu(x)}}\right),$$

so that $\mathcal{C}_1(x) \geq p$ is equivalent to

$$\frac{C - \mu(x)}{\sqrt{\lambda\mu(x)}} \geq \Phi^{-1}(p). \tag{7.12}$$

The left hand side of (7.12) is decreasing in $\mu(x)$, and thus $\mathcal{C}_1(x) \geq p$ is equivalent to $\mu(x) \leq \mu^*$, where μ^* is the unique root of the equation $C - \mu = \Phi^{-1}(p)\sqrt{\lambda\mu}$. □

We provide in Section 7.5.2 an application of Proposition 7.3 to a routing problem arising in telecommunications. Similar examples can be devised for the generalized assignment problem, see for instance the Proportional Mean-Variance Model from Spoerl and Wood (2003) which assumes that random variables are those from Proposition 7.3.

The next proposition considers the case of independent gamma random variables used, for instance, to model waiting and processing times in servers locations problems (Berman and Krass, 2004).

Proposition 7.4. *Consider n random variables $w_i \sim \Gamma(k_i, \theta)$, $i \in \mathsf{N}$, for some $\theta > 0$ and $k_i > 0$, and assume that $C > 0$. Then, if $x_i \in \{0,1\}$ for each $i \in \mathsf{N}$, the following constraints are equivalent:*

$$\mathcal{C}_1(x) \geq p \Leftrightarrow \sum k_i x_i \leq k^*,$$

where k^* is the unique solution of $\frac{\int_0^C z^{k-1} e^{\frac{-z}{\theta}} dz}{\Gamma(k)\theta^k} = p$ and the gamma function is defined by $\Gamma(k) = \frac{\int_0^\infty z^{k-1} e^{\frac{-z}{\theta}} dz}{\theta^k}$.

Proof. Gamma distributions satisfy also some kind of summation property, although weaker than the property satisfied by normal distributions. Recall that if $w_1, \dots, w_n$ are independent Gamma with shape k_i and a common scale θ, then $w := \sum w_i \sim \Gamma(k, \theta)$, with $k = \sum k_i$. Thus, if x_i are binary numbers, we have also that $w := \sum x_i w_i \sim \Gamma(k(x), \theta)$, with $k(x) = \sum k_i x_i$. Thus, for binary x_i, $\mathcal{C}_1(x)$ is equivalent to $P(\Gamma(k(x), \theta) \leq C)$ defined by

$$\frac{\int_0^C z^{k(x)-1} e^{\frac{-z}{\theta}} dz}{\Gamma(k(x)) \theta^{k(x)}},$$

which we note $\mathcal{K}(k(x))$ in the following. Then, assuming that $\mathcal{K}(k)$ is a strictly decreasing function of k, the constraint $\mathcal{K}(k(x)) \geq p$ is equivalent to the constraint $k(x) \leq k^*$, with $k^* = \mathcal{K}^{-1}(p)$ which proves $\mathcal{C}_1(x) \geq p \Leftrightarrow \sum k_i x_i \leq k^*$. Note that $\mathcal{K}^{-1}$ is well defined for any $p \in (0,1)$ because $\mathcal{K}$ is continuous, strictly decreasing, $\lim_{k \to 0^+} \mathcal{K}(k) = 1$ and $\lim_{k \to +\infty} \mathcal{K}(k) = 0$.

We are left to prove that $\mathcal{K}(k)$ is a strictly decreasing function of $k > 0$:

$$\begin{aligned}
\frac{d\mathcal{K}}{dk}(k) &= \theta \frac{d}{dk} \frac{\int_0^C z^{k-1} e^{-z} dz}{\int_0^\infty v^{k-1} e^{-v} dv} \\
&= \frac{\theta}{\Gamma^2(k)} \left(\int_0^C \ln(z) z^{k-1} e^{-z} dz \int_0^\infty v^{k-1} e^{-v} dv \right. \\
&\quad \left. - \int_0^C z^{k-1} e^{-z} dz \int_0^\infty \ln(v) v^{k-1} e^{-v} dv \right) \\
&= \frac{\theta}{\Gamma^2(k)} \int_0^C dz \int_0^\infty dv \left(z^{k-1} v^{k-1} e^{-z-v} (\ln(z) - \ln(v) \right) \\
&= \frac{\theta}{\Gamma^2(k)} \int_0^C dz \int_C^\infty dv \left(z^{k-1} v^{k-1} e^{-z-v} \ln \frac{z}{v} \right),
\end{aligned}$$

which is strictly negative because $\ln \frac{z}{v} < 0$ for $(z, v) \in [0, C] \times (C, \infty)$. □

When $C \leq 0$, $\mathcal{K}(k(x)) = 0$ so that the probabilistic constraint is equivalent to $\sum x_i \leq 0$.

7.5 Applications

In what follows we present two applications of the results from previous sections.

7.5.1 The chance-constraint knapsack

We apply Proposition 7.3 and 7.4 to a classical problem with a unique probabilistic constraint, studied by Klopfenstein and Nace (2008) and Goyal and Ravi (2009), among others. The next Corollary extends Theorem 5.9 from Chapter 5 to the chance-constraint knapsack problem under wider assumptions.

Corollary 7.5. *Let w_i be independent random variables, $i \in \mathsf{N}$, and assume that $C > 0$. Define the chance-constrained knapsack problem as follows:*

$$\begin{aligned} \max \quad & \sum p_i x_i \\ s.t. \quad & P\left(\sum w_i x_i \le C\right) \ge p \\ & x_i \in \{0,1\}. \end{aligned}$$

The following hold:

1. *If $w_i \sim \mathcal{N}(\mu_i, \lambda\mu_i)$, for some $\lambda > 0$ and a positive integer vector μ, the problem can be solved in $O\left(\sum \mu_i\right)$.*

2. *If $w_i \sim \Gamma(k_i, \theta)$, for some $\theta > 0$ and a positive integer vector k, the problem can be solved in $O\left(\sum k_i\right)$.*

Proof. Consider case *2.*, when $w_i \sim \Gamma(k_i, \theta)$. Using Proposition 7.4, we can replace the probabilistic constraint by a linear one with integer coefficients, making the problem a knapsack problem with complexity $O\left(\sum k_i\right)$. We must only check that the new capacity, k^* can be computed in $O\left(\sum k_i\right)$.

Because k is integer and x binary, $\sum k_i x_i \le k^*$ is equivalent to $\sum k_i x_i \le \lfloor k^* \rfloor$ so that we only need to compute $C := \lfloor k^* \rfloor$. Therefore, we describe next how to compute C in $O\left(\log \sum_{i=1}^n k_i\right)$. First, if $\mathcal{K}\left(\sum k_i\right) \ge p$, then $k^* \ge \sum k_i$ and we can set $C := \sum k_i$. Otherwise, compute $\lfloor k^* \rfloor$ by a dichotomic search based on the sign of $\mathcal{K}(k) - p$. Case *1.* is proved similarly. □

Note that similar results can be obtained from Propositions 7.1 and 7.2, although the complexity of computing $P\left(\sum w_i \le C\right)$ depends on the specific distribution of w.

7.5.2 The bandwidth packing problem

In what follows, we apply Proposition 7.3 to a multi-commodity flow problem occurring in telecommunications networks. We discuss different approaches to tackle the probabilistic constraints. Notice that our example is easily extended to the problem of designing a telecommunications network, introducing binary design vector y and replacing the fixed capacity C_{ij} by $C_{ij} y_{ij}$.

Problem description

Given a directed graph $\mathsf{G} = (\mathsf{V}, \mathsf{A})$ with a capacity vector C, and a set of commodities Q of size d^q and revenue c^q from $s(q)$ to $t(q)$ for each $q \in \mathsf{Q}$, the bandwidth packing problem (**BWP**) aims at routing commodities on the network in order to maximize the total revenue. For technical reasons based on routing protocols, each commodity must be sent along a unique path from $s(q)$ to $t(q)$, see Barnhart et al. (2000); Park et al. (1996). Introducing the binary variable x_{ij}^q stating whether commodity q is routed through arc (i,j), the problem can be formulated as

$$\begin{aligned} \max \quad & \sum_{q \in \mathsf{Q}} c^q \left(\sum_{i \in \mathsf{V}:(i,t(q)) \in \mathsf{A}} x_{it(q)}^q \right) & & \\ & \sum_{q \in \mathsf{Q}} d^q x_{ij}^q \leq C_{ij} & (i,j) \in \mathsf{A} & \qquad (7.13) \\ \text{s.t.} \quad & \sum_{j \in \mathsf{V}:(j,i) \in \mathsf{A}} x_{ji}^q - \sum_{j \in \mathsf{V}:(i,j) \in \mathsf{A}} x_{ij}^q = 0 & i \in \mathsf{V}, q \in \mathsf{Q} & \\ & x \text{ binary.} & & \end{aligned}$$

In practice, although the traffic size d^q varies along time, it is not convenient to change the routing according tho these variations; x must be set once for a given time period. Different frameworks allow to model such uncertainties. Some works consider that d belongs to a polyhedron $\mathcal{D}$ and that (7.13) must be feasible for any $d \in \mathcal{D}$, see Klopfenstein and Nace (2010) and the closely related Altin et al. (2007), among others. Others (Klopfenstein, 2009; Pascali, 2009) model d^q, $q \in \mathsf{Q}$, by random variables and replace (7.13) by

$$P\left(\sum_{q \in \mathsf{Q}} d^q x_{ij}^k \leq C_{ij} \right) \geq p \qquad (i,j) \in \mathsf{A}. \qquad (7.14)$$

In what follows, we assume that d^q, $q \in \mathsf{Q}$, are independent Gaussian distributed according to $\mathcal{N}(\mu^k, \lambda\mu^k)$. The Gaussian assumption has been studied by Alagöz (2002) and Kilpi and Norros (2002) and used by Andrade et al. (2004) and Klopfenstein (2009), among others. Moreover, Sen et al. (1994) (followed by Lisser et al. (1999) and Andrade et al. (2004)) assume that d^q and d^h are independently distributed for $k \neq h$. Finally, Morris and Lin (2000) suggest that means and variances are linearly correlated as traffic size increases, that is, $\sigma = \lambda\mu$ for some $\lambda > 0$, so that we can apply Proposition 7.3 to (7.14).

Solution methods

We review different approaches to tackle the chance-constrained version of (**BWP**). Besides Proposition 7.3, there are two groups of methods to handle (7.14). Keeping the random vector continuous, we can tackle (7.14) by MINLP methods. Alternatively, we can sample the random variables to obtain a scenario set S and solve the deterministic equivalent.

Direct linearization We apply Proposition 7.3 to (7.14), obtaining again problem (**BWP**) with d^q and C_{ij} replaced by μ^q and the unique root μ_{ij}^* of $C_{ij} - \mu = \Phi^{-1}(p)\sqrt{\lambda\mu}$, respectively. Computing μ_{ij}^* is easy since function $\frac{C_{ij}-\mu}{\sqrt{\lambda\mu}}$ is convex and differentiable. Therefore, we can solve the problem with efficient algorithms used in the deterministic case, such as the branch-and-cut-and-price algorithm from Barnhart et al. (2000).

MINLP methods When $p \geq 0.5$ and each d^q is Gaussian, (7.14) is convex and thus, well suited for non-linear algorithms Bonami et al. (2009). However, it is clearly easier to use the direct linearization of (7.14) through Proposition 7.3, because non-linear constraints are harder to handle than linear ones and both formulations provide the same bound. For instance, outer approximation-based algorithms replace (7.14) by a set of tangent cutting planes. The latter contains more inequalities, with possibly highly fractional coefficients, than the unique inequality resulting from (7.14).

Alternatively, (7.14) with Gaussian random variables can be reformulated as (7.10). We can then rewrite (7.10) as

$$\sum_{q \in \mathsf{Q}} \mu^q x_{ij}^q \leq C_{ij} \tag{7.15}$$

$$\sum_{q \in \mathsf{Q}} \left[(\Phi^{-1}(p)\sigma^q)^2 + \mu^q(2C_{ij} - \mu^q) \right] x_{ij}^q + \sum_{q_1, q_2 \in \mathsf{Q} : q_1 \neq q_2} \left[((\Phi^{-1}(p))^2 \sigma^{q_1}\sigma^{q_2} - 2\mu^{q_1}\mu^{q_2}) \right] x_{ij}^{q_1} x_{ij}^{q_2} \leq C_{ij}^2, \tag{7.16}$$

for each $(i,j) \in \mathsf{A}$. When $p > 0.5$, which is the case in real situations, $\mu_{ij}^* < C_{ij}$ and thus, (7.15) is less tight than (7.11). Hence, Proposition (7.14) allows to strengthen the above formulation by substituting (7.15) with (7.11). Then, (7.16) is not needed anymore to define a valid formulation. However, since it takes into account the binary restriction on x (by using $(x_{ij}^q)^2 = x_{ij}^q$), it may be used together with (7.11) to provide a stronger continuous relaxation. Note finally that linearizing (7.16) requires at least $|\mathsf{Q}|$ additional variables and $2|\mathsf{Q}|$ additional constraints for each $(i,j) \in \mathsf{A}$, see Hansen and Meyer (2009).

Discretization and deterministic equivalent Sampling a scenario set Ω that approximates the continuous distribution d in an acceptable way, see Luedtke and Ahmed (2008) and Pagnoncelli et al. (2009), among others, we can write a deterministic equivalent for (7.14):

$$\sum_{q \in \mathsf{Q}} d^q(\omega) x_{ij}^q \leq C_{ij} + M_{ij}(\omega)(1 - z_{ij}(\omega)) \qquad (i,j) \in \mathsf{A}, \omega \in \Omega \tag{7.17}$$

$$\sum_{\omega \in \Omega} p(\omega) z_{ij}(\omega) \geq p \qquad (i,j) \in \mathsf{A} \tag{7.18}$$

$$z \text{ binary},$$

where components of vector M are numbers large enough. However, (7.17) and (7.18) yield a very difficult problem because (7.17) contains a large number of constraints and features “big-M” coefficients. Therefore, Beraldi and Bruni (2009, 2010) show how to replace (7.17) and (7.18) by a relevant set L of scenario sets through a branch-and-bound algorithm. Each $l \in \mathsf{L}$ yields a problem similar to (**BWP**), but with multiple capacity constraints (7.13) for each arc $(i, j) \in \mathsf{A}$ (one for each scenario in l). Then, using bounding mechanisms, they avoid solving all problems associated to elements of L. Eventually, the exact approaches from Beraldi and Bruni (2009, 2010) will have solved several binary multi-commodity flow problems with multiple capacity constraints, each of them being more more complex than (**BWP**). Although applicable to a broader class of problems, this approach will in general be slower than the direct linearization from Proposition 7.3 that requires only to solve one problem similar to (**BWP**) plus the computation of the root vector μ^*.

Conclusion and perspectives

In what follows, we conclude by giving a short summary of this thesis. For each chapter, we review and criticize the work done. Directions for future research are also proposed. Among them, we think that the most promising ones are related to Chapters 4 and 6.

Network design problems

The first part of this thesis presented network design problems applied to telecommunications and power transmission networks.

Benders decomposition for telecommunications network design

In Chapter 3, we studied two models that arise in telecommunications, the bi-layer network design and the hop-constrained path diversified network design. Because both models handle multi-commodity flows, they lead to large-scale formulations for which decomposition is recommended. We devised a Benders decomposition for each problem, and implemented various cutting-plane and branch-and-cut algorithms to generate Benders cuts. We conducted a thorough computational study on a large set of instances. For the bi-layer network design problem, our branch-and-cut algorithm outperforms the cutting plane algorithm of Knippel and Lardeux (2007) by a factor of 10 in average.

For the model (**ML**), our main goal was to show that branch-and-cut algorithms can be an order of magnitude faster than cutting plane algorithms if they are implemented correctly. Although our tests were applied to a multi-layer network design problem, we believe that many Benders decompositions reported in the literature could be enhanced by using branch-and-cut algorithms. On a second model, (**HOP**), we investigate further at which point the Benders cuts should be generated in the course of the algorithm.

Our results could be improved by a study of strong cutting planes. For (**ML**), it could be interesting to test well known strong cutting planes for the single layer network design problems. For (**HOP**), finding new strong cutting planes could be a subject for future research. Furthermore, although a very efficient heuristic has been devised for (**HOP**), we have not yet tested for one in the (**ML**) case.

Transmission expansion planning with re-design

Chapter 4 studied the problem of expanding a power transmission network. We considered a linearized version of Kirchoff laws because it provides a realistic approximation of physical laws governing power flows while being computationally tractable. We showed on an example that, because flows are governed by these physical laws, a power network may become more efficient after cutting off some of its circuits. For this reason, we introduced a new model which allows for cutting off some circuits when expanding the network. The linearized Kirchoff laws yield bilinear constraints which are linearized by introducing "big-M" coefficients. We proved that the computation of minimal values for the "big-M" coefficients requires to look for shortest and longest paths in the network. We compared numerically different formulations for the problem, including a new one. Finally, we conducted numerical experiments on real-based networks, and showed that our re-design model may yield significant cost reductions.

Our comparison of the formulations is only empirical. It would be very interesting to also conduct a theoretical study, to find out whether these formulations always yield the same LP relaxation. Also, we were disappointed by the poor results obtained by our new compact formulation within CPLEX. We have yet been unable to understand why, however, more numerical results or examples could help to answer this question.

Then, survivability was briefly mentioned in Chapter 4, by mean of $(N-1)$ constraints. In addition to $(N-1)$ constraints, it is important for the expansion planning problem to consider uncertainty both in the electricity demand and generation. In this case, instead of (or in addition to) considering contingency scenarios $h \in \mathsf{E}$, the 2-stage formulation $(\mathbf{TEP_R} - \mathbf{N1})$ makes use of a set Ω such that to each scenario $\omega \in \Omega$ corresponds a demand/generation vector $(d(\omega), G(\omega))$. As in $(\mathbf{TEP_R} - \mathbf{N1})$, the design decisions y must be taken *here-and-now*, while the *wait-and-see* decisions of recourse $(x(\omega), g(\omega), u(\omega), \theta(\omega))$ depend on each scenario $\omega \in \Omega$. The main difference with the reliability models is that demand/generation scenarios do not need the additional vector δ, because uncertainty is fully characterized by the values of $d(\omega)$ and $G(\omega)$. Finally, one could consider both the $(N-1)$ reliability criterion and different scenarios for demand and generation (Oliveira et al., 2007). For networks bigger than "Garver", solving to optimality $(\mathbf{TEP_R} - \mathbf{N1})$ or one of the extensions mentioned above requires that efficient decomposition algorithms be developed which would constitute an interesting subject for future research.

Stochastic models

The second part of this thesis studied stochastic models. More specifically, we studied two-stage stochastic programming with simple recourse and chance-constrained programming.

The stochastic knapsack problem with simple recourse

Chapter 5 studies the stochastic knapsack problem with simple recourse. We first reformulate the problem as a convex MINLP. Using a branch-and-cut algorithm inspired by the LP/NLP algorithm, we can solve efficiently the problem for thousands of variables, provided the random variables are Gaussian. Then we study the complexity of the problem and find out three particular cases that we prove to be weakly $\mathcal{NP}$-complete.

Our numerical results only consider Gaussian random variables. However, using packages to perform numerical integration such as (Prékopa, 1995), we could have tested our branch-and-cut algorithm with more general random variables. It would be interesting to know whether the problem remains as simple as in the Gaussian case. Also, we did not compute the quality of the NLP relaxation of our MINLP. Since the number of generated cutting planes is low, we believe that the gap is also small.

Chapter 6 and 7 are based on ideas developed in Chapter 5. Further directions could seek to extend the complexity results for other random variables, and to the case of integer recourse.

Dantzig-Wolfe decomposition for MINLP applied to stochastic network design

In Chapter 6 we specialized the branch-and-cut algorithm from Chapter 5 to handle a stochastic network design problem with simple recourse. First, we formulate the problem as a convex MINLP linearly constrained. The resulting formulation has a large number of constraints, such that we thought of developing a decomposition algorithm. We chose to experiment the Dantzig-Wolfe decomposition because it is more flexible than the Benders decomposition. Moreover, it had never been used for MINLP before. We implemented a naive version of a branch-and-cut-and-price algorithm obtaining good preliminary results.

We see two directions of improvement for this work. First, we should implement our branch-and-cut-and-price framework within an existing framework, such as SCIP, in order to ease the incorporation of advanced techniques such as cutting planes, heuristics, preprocessing and constraints propagation, among others. Second, we believe Dantzig-Wolfe decomposition can provide interesting results for a larger class of MINLP problems, similarly to the good results it has already given for MIP.

Easy distributions for combinatorial optimization problems with probabilistic constraints

In Chapter 7, we applied to individual probabilistic constraints a technique similar to the one used to prove Theorem 5.9. This allows us to linearize such constraints when all variables are binary and all random coefficients are independently distributed according to either $\mathcal{N}(\mu_i, \lambda\mu_i)$, for some $\lambda > 0$ and $\mu_i > 0$, or $\Gamma(k_i, \theta)$ for some $\theta > 0$ and $k_i > 0$. The constraint can also be linearized when the coefficients are independent and identically distributed and either positive or strictly stable random variables.

Although the results are very strong because they make hard chance-constrained problems as easy as their deterministic counterpart, it is not clear where one can find practical applications involving random variables that satisfy the hypotheses above. Future work should try to apply these results to realistic problems, or to extend them to more general random variables.

Bibliography

K. Abhishek, S. Leyffer, and J. T. Linderoth. Filmint: An outer approximation-based solver for convex mixed-integer nonlinear programs. *INFORMS JOURNAL ON COMPUTING.* In press. [cited on p. 54]

K. Abhishek, S. Leyffer, and J. T. Linderoth. Filmint: An outer-approximation-based solver for nonlinear mixed integer programs. *INFORMS Journal on computing*, 2010. In press. [cited on p. 83]

T. Achterberg. SCIP: Solving constraint integer programs. *Math. Programming Comput.*, 1:1–41, 2009. [cited on p. 31]

T. Achterberg and C. Raack. The MCF-separator – detecting and exploiting multi-commodity flows in MIPs. *Mathematical Programming C*, (2):125–165, 2010. [cited on p. 12 and 33]

R. K. Ahuja, T. L. Magnanti, and James B. Orlin. *Network flows, theory, algorithms ans applications.* Prentice Hall, 1993. [cited on p. 53 and 61]

F. Alagöz. Approximations on the aggregate mpeg video traffic and their impact on admission control. *Turk. J. Elec. Eng.*, 10(1):73–84, 2002. [cited on p. 112]

N. Alguacil, A. L. Motto, and A. J. Conejo. Transmission expansion planning: A mixed-integer lp approach. *IEEE Transactions on Power Systems*, 18(3):1070–1077, 2003. [cited on p. 64]

A. Altin, Edoardo Amaldi, Pietro Belotti, and Mustafa Ç. Pinar. Provisioning virtual private networks under traffic uncertainty. *Networks*, 49(1):100–115, 2007. [cited on p. 95 and 112]

A. Altin, H. Yaman, and M. Ç. Pinar. The robust network loading problem under polyhedral demand uncertainty: Formulation, polyhedral analysis and computations. *INFORMS J. Comput.*, 2010. in press. [cited on p. 95]

R. Andrade, A. Lisser, N. Maculan, and G. Plateau. Telecommunication network capacity design for uncertain demand. *Comput. Optim. Appl.*, 29(2):127–146, 2004. ISSN 0926-6003. [cited on p. 112]

R. Andrade, A. Lisser, N. Maculan, and G. Plateau. B&b frameworks for the capacity expansion of high speed telecommunication networks under uncertainty. *Ann. Oper. Res.*, 150(1):49–65, 2005. [cited on p. 94]

A. Atamturk. On capacitated network design cut-set polyhedra. *Math. Programming*, 92:425–437, 2002. [cited on p. 12]

P. Avella, S. Mattia, and A. Sassano. Metric inequalities and the network loading problem. *Discrete Optimization*, 4:103–114, 2007. [cited on p. 30]

T. Badics and E. Boros. Minimization of half-products. *Math. Oper. Res.*, 23(3):649–660, 1998. [cited on p. 78]

L. Bahiense, G.C. Oliveira, M. Pereira, and S. Granville. A mixed integer disjunctive model for transmission network expansion. *Power Systems, IEEE Transactions on*, 16(3):560–565, Aug 2001. [cited on p. 46, 54, and 56]

G. Baier, T. Engel, A. M. C. A. Koster, S. Orlowski, C. Raack, and R. Wessäly. Single-layer cuts for multi-layer network design problems. ZIB Report ZR-07-21, August 2007. [cited on p. 22]

A. Balakrishnan and K. Altinkemer. Using a hop-constrained model to generate alternative communication network design. *ORSA J. Comput.*, 4:192–205, 1992. [cited on p. 23]

C. Barnhart, E. L. Johnson, G. L. Nemhauser, Martin W. P. Savelsbergh, and Pamela H. Vance. Branch-and-price: column generation for solving huge integer programs. *Oper. Res.*, 46(3): 316–329, 1998. [cited on p. 91]

C. Barnhart, C. A. Hane, and P. H. Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Oper. Res.*, 48(2):318–326, 2000. [cited on p. 13, 73, 98, 111, and 113]

P. Belotti and F. Malucelli. Row-column generation for multilayer network design. In *Proceedings, International Network Optimization Conference, 2005, Lisbon, Portugal*, March 2005. [cited on p. 22]

P. Belotti, A. Capone, G. Carello, F. Malucelli, F. Senaldi, and A. Totaro. Mpls over transport network: Two layers approach to network design with statistical multiplexing. In *Conference on Next Generation Internet Design and Engineering (NGI 2006), Valencia (Spain)*, 2006. [cited on p. 22]

W. Ben-Ameur. Between fully dynamic routing and robust stable routing. In *6th International Workshop on Design and Reliable Communication Networks, 2007. DRCN 2007.* [cited on p. 95]

A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Math. Program.*, 99(2):351–376, 2004. [cited on p. 94]

J. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962. [cited on p. 26]

P. Beraldi and M. E. Bruni. A probabilistic model applied to emergency service vehicle location. *European Journal of Operational Research*, 196(1):323–331, 2009. [cited on p. 104, 113, and 114]

P. Beraldi and M. E. Bruni. An exact approach for solving integer problem under probabilistic constraints with random technology matrix. *Annals of Operations Research*, 177:127–137, 2010. [cited on p. 104, 113, and 114]

P. Beraldi and A. Ruszczynski. The probabilistic set-covering problem. *Oper. Res.*, 50(6):956–967, 2002a. [cited on p. 103 and 104]

P. Beraldi and A. Ruszczynski. A branch and bound method for stochastic integer problems under probabilistic constraints. *Optimization Methods and Software*, 17:359–382, 2002b. [cited on p. 103]

P. Beraldi and A. Ruszczynski. Beam search heuristic to solve stochastic integer problems under probabilistic constraints. *European Journal of Operational Research*, 167(1):35–47, 2005. [cited on p. 103]

Bibliography

O. Berman and D. Krass. *Facility location problems with stochastic demands and congestion*, chapter 11, pages 329–371. In: Drezner, Z. and Hamacher, H.W. (eds.) Facility location: applications and theory. Springer, 2004. [cited on p. 109]

D. Bienstock and O. Gunluk. Capacitated Network Design–Polyhedral Structure and Computation. *INFORMS JOURNAL ON COMPUTING*, 8(3):243–259, 1996. [cited on p. 12]

D. Bienstock and S. Mattia. Using mixed-integer programming to solve power grid blackout problems. *Discrete Optimization*, 4(1):115–141, 2007. [cited on p. 51]

D. Bienstock, S. Chopra, O. Günlük, and C.-Y. Tsai. Minimum cost capacity installation for multicommodity network flows. *Math. Program.*, 81:177–199, 1998. [cited on p. 12]

S. Binato. Optimal expansion of transmission networks by benders decomposition and cutting planes. Ph.D. dissertation (Portuguese), Federal University of Rio de Janeiro, 2000. [cited on p. 62 and 64]

S. Binato, G. C. Oliveira, and J. L.Araújo. A Greedy Randomized Adaptive Search Procedure for Transmission Expansion Planning. *IEEE Transactions on Power Systems*, 16(2):247–253, 2001. [cited on p. 50]

J. R. Birge and F. V. Louveaux. *Introduction to Stochastic programming (2nd edition).* Springer Verlag, New-York, 2008. [cited on p. 27, 32, 46, 67, and 75]

P. Bonami, L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, and A. Wächter. An algorithmic framework for convex mixed integer nonlinear programs. *Discrete Optimization*, 5(2):186–204, 2008. [cited on p. 83]

P. Bonami, M. Kilinc, and J. Linderoth. Algorithms and software for convex mixed integer nonlinear programs. Technical Report 1664, Computer Sciences Department, University of Wisconsin-Madison, 2009. [cited on p. 92, 97, and 113]

P. Bonami, M. Kilinc, and J. Linderoth. *IMA Volumes.* University of Minnesota, 2010. Algorithms and Software for Convex Mixed Integer Nonlinear Programs. [cited on p. 83]

Q. Botton. *Survivable network design with quality of service constraints.* PhD thesis, Université catholique de Louvain, Louvain School of Management, 2010. [cited on p. 21, 27, and 33]

O. Briant, C. Lemaréchal, P. Meurdesoif, S. Michel, N. Perrot, and F. Vanderbeck. Comparison of bundle and classical column generation. *Math. Program.*, 113(2):299–344, 2008. [cited on p. 91]

M.O. Buygi, G. Balzer, H.M. Shanechi, and M. Shahidehpour. Market based transmission expansion planning: fuzzy risk assessment. volume 2, pages 427–432 Vol.2, April 2004. [cited on p. 46]

M.O. Buygi, H.M. Shanechi, G. Balzer, M. Shahidehpour, and N. Pariz. Network planning in unbundled power systems. *Power Systems, IEEE Transactions on*, 21(3):1379–1387, Aug. 2006. [cited on p. 46]

G. Calafiore and L. El Ghaoui. Distributionally robust chance-constrained linear programs with applications. *J. of Optimization Theory and Applications*, 130(1):1–22, 2006. [cited on p. 104]

A. Capone, G. Carello, and R. Matera. Multi-layer network design with multicast traffic and statistical multiplexing. In *IEEE GLOBECOM 2007, Washington DC, USA*, 2007. [cited on p. 22]

C. C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Math. Programming*, 83(3, Ser. A):451–464, 1998. [cited on p. 27]

J. Choi, T. Tran, A.A. El-Keib, R. Thomas, H. Oh, and R. Billinton. A method for transmission system expansion planning considering probabilistic reliability criteria. *Power Systems, IEEE Transactions on*, 20(3):1606–1615, Aug. 2005. [cited on p. 46]

J. Choi, T.D. Mount, and R.J. Thomas. Transmission expansion planning using contingency criteria. *Power Systems, IEEE Transactions on*, 22(4):2249–2261, Nov. 2007. [cited on p. 46]

V. Chvátal. Hard knapsack problems. *Operations Research*, 28:1402–1411, 1980. [cited on p. 85]

G. Codato and M. Fischetti. Combinatorial benders' cuts for mixed-integer linear programming. *Oper. Res.*, 54(4):756–766, 2006. [cited on p. 33]

A. Cohn and C. Barnhart. The stochastic knapsack problem with random weights: A heuristic approach to robust transportation planning. In *Proceedings of the Triennial Symposium on Transportation Analysis (TRISTAN III)*, 1998. [cited on p. 74 and 80]

A. Costa, J.-F. Cordeau, and B. Gendron. Benders, metric and cutset inequalities forămulticommodity capacitated networkădesign. *Computational Optimization and Applications*, 42:371–392, 2009. [cited on p. 26]

A. M. Costa. A survey on benders decomposition applied to fixed-charge network design problems. *Comput. Oper. Res.*, 32(6):1429–1450, 2005. [cited on p. 13 and 26]

T. G. Crainic. Service network design in freight transportation. *European Journal of Operational Research*, 122:272–288, 2000. [cited on p. 12 and 48]

T. G. Crainic, A. Frangioni, and B. Gendron. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112(1-3):73–99, 2001. [cited on p. 13]

G. Dahl. Notes on polyhedra associated with hop-constrained walk polytopes. *Oper. Res. Lett.*, 25:97–100, 1999. [cited on p. 23]

G. Dahl and M. Stoer. A cutting plane algorithm for multicommodity survivable network design problems. *INFORMS J. on Computing*, 10(1):1–11, 1998. [cited on p. 12]

G. Dahl, A. Martin, and Mechthild Stoer. Routing through virtual paths in layered telecommunication networks. *Oper. Res.*, 47(5):693–702, 1999. [cited on p. 22]

G. Dahl, N. Foldnes, and L. Gouveia. A note on hop-constrained walk polytopes. *Oper. Res. Lett.*, 32(4):345–349, 2004. [cited on p. 23]

I. de J. Silva Junior. Planejamento da expansao de sistemas de transmissao considerando seguranca e planos de programacao da geracao. Ph.D. dissertation (Portuguese), Universidade Estadual de Campinas, 2005. [cited on p. 46 and 64]

S. de la Torre, A.J. Conejo, and J. Contreras. Transmission expansion planning in electricity markets. *Power Systems, IEEE Transactions on*, 23(1):238–248, Feb. 2008. [cited on p. 46]

D. Dentcheva, A. Prékopa, and A. Ruszczynski. Bounds for probabilistic integer programming problems. *Discrete Applied Mathematics*, 124(1-3):55 – 65, 2002. [cited on p. 103]

E.J. deOliveira, Jr. daSilva, I.C., J.L.R. Pereira, and Jr. Carneiro, S. Transmission system expansion planning using a sigmoid function to handle integer investment variables. *Power Systems, IEEE Transactions on*, 20(3):1616–1621, Aug. 2005. [cited on p. 46]

M. Deza and M. Laurent. *Geometry of Cuts and Metrics*, volume 15. Springer, 1997. [cited on p. 29]

E. D. Dolan and J. J. More. Benchmarking optimization software with performance profiles. *Math. Programming*, 91(2):201–213, 2002. [cited on p. 38]

M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Program.*, 36(3):307–339, 1986. [cited on p. 75]

R. Fang and D.J. Hill. A new strategy for transmission expansion in competitive electricity markets. *Power Systems, IEEE Transactions on*, 18(1):374–380, Feb 2003. [cited on p. 46]

M. Fischetti, D. Salvagnin, and A. Zanette. A note on the selection of benders cuts. *Math. Program.*, 124(1-2):175–182, 2010. ISSN 0025-5610. [cited on p. 20 and 28]

B. Fortz and M. Poss. An improved benders decomposition applied to a multi-layer network design problem. *Oper. Res. Lett.*, 37:359–364, 2009. [cited on p. 21]

B. Fortz, Q. Botton, L. Gouveia, and M. Poss. Benders decomposition for the hop-constrainted survivable network design problem. Technical report, GOM, Université Libre de Bruxelles, 2010. [cited on p. 21, 27, and 33]

A. Frangioni and B. Gendron. 0-1 reformulations of the multicommodity capacitated network design problem. *Discrete Appl. Math.*, 157(6):1229–1241, 2009. [cited on p. 13]

A. Frangioni and B. Gendron. A stabilized structured dantzig- wolfe decomposition method. Technical Report CIRRELT-2010-02, 2010. [cited on p. 13]

V. Gabrel, A. Knippel, and M. Minoux. Exact solution of multicommodity network optimization problems with general step cost functions. *Oper. Res. Lett.*, 25:15–23(9), 1999. [cited on p. 13, 22, and 32]

M.R. Garey and D.S. Johnson. *Computers and intractability. A guide to the theory of NP-completeness. A Series of Books in the Mathematical Sciences.* WH Freeman and Company, San Francisco, Calif, 1979. [cited on p. 8]

L.L. Garver. Transmission network estimation using linear programming. *IEEE Trans. Power Appar. Syst.*, 89(7):1688– 1697, 1970. [cited on p. 51 and 64]

J. Geffard, B. Lardeux, and D. Nace. Multiperiod network design with incremental routing. *Netw.*, 50(1):109–117, 2007. [cited on p. 22]

L. Gouveia. Multicommodity flow models for spanning trees with hop constraints. *Eur. J. Oper. Res.*, 95:178–190, 1996. [cited on p. 36]

L. Gouveia. Using variable redefinition for computing lower bounds for minimum spanning and steiner trees with hop constraints. *INFORMS J. Comput.*, 10:180–188, 1998. [cited on p. 21, 23, and 25]

L. Gouveia and T. L. Magnanti. Network flow models for designing diameter-constrained minimum-spanning and steiner trees. *Networks*, 41(3):159–173, 2003. [cited on p. 23]

L. E. Gouveia, P.F. Patrício, A.F. de Sousa, and R. Valadas. MPLS over WDM Network Design with Packet Level QoS Constraints based on ILP Models. In *Proceedings of IEEE Infocom*, 2003. [cited on p. 23]

V. Goyal and R. Ravi. Chance constrained knapsack problem with random item sizes. Submitted to Oper. Res. Lett., 2009. [cited on p. 110]

C. Grinstead and J. Snell. *Introduction to Probability, 2nd edn.* American Math. Society, Providence, 1997. [cited on p. 107]

I. E. Grossmann. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimization and Engineering*, 3:227–252(26), 2002. [cited on p. 10 and 103]

P. Hansen and C. Meyer. Improved compact linearizations for the unconstrained quadratic 0–1 minimization problem. *Discrete Applied Mathematics*, 157:1267–1290, 2009. [cited on p. 104, 109, and 113]

B. Hansotia. Some special cases of stochastic programs with recourse. *Operations Research*, 25(2): 361–363, 1977. [cited on p. 82]

W. W. Hardgrave and G. L. Nemhauser. On the relation between the traveling-salesman and the longest-path problems. *Oper. Res.*, 10(5):647–657, 1962. [cited on p. 59 and 64]

R. Henrion. Introduction to chance-constrained programming. Tutorial paper for the Stochastic Programming Community Home Page, 2004. [cited on p. 103]

R. Henrion and C. Strugarek. Convexity of chance constraints with independent random variables. *Comput. Optim. Appl.*, 41:263–276, 2008. [cited on p. 103]

H. Holler and S. Voss. A heuristic approach for combined equipment-planning and routing in multi-layer sdh/wdm networks. *Eur. J. Oper. Res.*, 127(3):787–796, June 2006. [cited on p. 22]

D. Huygens and A. R. Mahjoub. Integer programming formulations for the two 4-hop-constrained paths problem. *Networks*, 49(2):135–144, 2007. [cited on p. 23]

D. Huygens, A. R. Mahjoub, and P. Pesneau. Two edge-disjoint hop-constrained paths and polyhedra. *SIAM J. Discrete Math.*, 18(2):287–312, 2004. [cited on p. 23]

D. Huygens, M. Labbé, A. R. Mahjoub, and P. Pesneau. The two-edge connected hop-constrained network design problem: Valid inequalities and branch-and-cut. *Networks*, 49(1):116–133, 2007. [cited on p. 21 and 23]

IBM-ILOG. IBM-ILOG Cplex, 2009. http://www.ilog.com/products/cplex/. [cited on p. 31, 65, and 84]

R. G. Jeroslow. There cannot be any algorithm for integer programming with quadratic constraints. *Op. Res.*, 21:221–224, 1973. [cited on p. 8]

R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972. [cited on p. 53]

S. Kataoka. A stochastic programming model. *Econometrica*, 31(1/2):181–196, 1963. [cited on p. 104]

J. Kilpi and I. Norros. Testing the gaussian approximation of aggregate traffic. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 49–61, New York, NY, USA, 2002. ACM. ISBN 1-58113-603-X. [cited on p. 112]

A. J. Kleywegt and J. D. Papastavrou. The dynamic and stochastic knapsack problem with random sized items. *Oper. Res.*, 49(1):26–41, 2001. [cited on p. 75]

A. J. Kleywegt, A. Shapiro, and T. Homem de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM J. Optim.*, 12(2):479–502, 2002. [cited on p. 74, 75, and 94]

O. Klopfenstein. Dimensionnement de réseau avec prévisions de demandes incertaines et contrainte de monoroutage. In *proc. of Algotel, 2009 (Carry-le-Rouet, France)*, 2009. [cited on p. 112]

O. Klopfenstein. Solving chance-constrained combinatorial problems to optimality. *Computational Optimization and Applications*, 45(3):607–638, 2010. [cited on p. 104]

O. Klopfenstein and D. Nace. Valid inequalities for a robust knapsack polyhedron - application to the robust bandwidth packing problem. *Networks*. In press. [cited on p. 73]

O. Klopfenstein and D. Nace. A note on polyhedral aspects of a robust knapsack problem. Optimization Online, 2007. [cited on p. 75]

O. Klopfenstein and D. Nace. A robust approach to the chance-constrained knapsack problem. *Oper. Res. Lett.*, 36(5):628–632, 2008. [cited on p. 75 and 110]

O. Klopfenstein and D. Nace. Valid inequalities for a robust knapsack polyhedron - application to the robust bandwidth packing problem. *Networks*, 2010. to appear. [cited on p. 112]

A. Knippel and B. Lardeux. The multi-layered network design problem. *Eur. J. Oper. Res.*, 127 (1):87–99, November 2007. [cited on p. 21, 22, 29, 37, and 115]

A. Knippel, B. Lardeux, and J. Geffard. Efficient algorithms for solving the 2-layered network design problem. In *Proceedings of INOC, Paris*, 2003. [cited on p. 22]

A. M. C. A. Koster, S. Orlowski, C. Raack, and R. Wessäly. Two-layer network design by branch-and-cut featuring MIP-based heuristics. In *Proceedings of INOC, Spa, Belgium*, April 2007. [cited on p. 22]

S. Kosuch and A. Lisser. Upper bounds for the 0-1 stochastic knapsack problem and a b&b algorithm. *Ann. Oper. Res.*, 2009. Article in press. [cited on p. 74]

E. Kubilinskas and M. Pióro. An ip/mpls over wdm network design problem. In *In Proceedings of INOC, Lisbon*, volume 3, 2005. [cited on p. 22]

G. Laporte, F. V. Louveaux, and L. Van hamme. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Oper. Res.*, 50(3):415–423, 2002. [cited on p. 75]

G. Latorre, R.D. Cruz, J.M. Areiza, and A. Villegas. Classification of publications and models on transmission expansion planning. *Power Systems, IEEE Transactions on*, 18(2):938–946, May 2003. [cited on p. 46]

M. A. Lejeune and A. Ruszczynski. An efficient trajectory method for probabilistic production-inventory-distribution problems. *Operations Research*, 55(2):378–394, 2007. [cited on p. 103]

A. Lisser, A. Ouorou, J.-P. Vial, and J. Gondzio. Capacity planning under uncertain demand in telecommunication networks. Technical report, 1999. [cited on p. 94 and 112]

I. Ljubic, P. Putz, and J.J. Salazar. Exact approaches to the single-source network loading problem. Technical Report 2009-05, University of Vienna, 2009. [cited on p. 13, 20, and 28]

J.A. Lopez, K. Ponnambalam, and V.H. Quintana. Generation and transmission expansion under risk using stochastic programming. *Power Systems, IEEE Transactions on*, 22(3):1369–1378, Aug. 2007. [cited on p. 46]

M. Lu, Z.Y. Dong, and T.K. Saha. A framework for transmission planning in a competitive electricity market. In *Transmission and Distribution Conference and Exhibition: Asia and Pacific, 2005 IEEE/PES*, pages 1–6, 2005. [cited on p. 46]

M. E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Oper. Res.*, 53(6):1007–1023, 2005. [cited on p. 91]

J. Luedtke and S. Ahmed. A sample approximation approach for optimization with probabilistic constraints. *SIAM Journal on Optimization*, 19(2):674–699, 2008. [cited on p. 113]

P. Maghouli, S.H. Hosseini, M.O. Buygi, and M. Shahidehpour. A multi-objective framework for transmission expansion planning in deregulated environments. *Power Systems, IEEE Transactions on*, 24(2):1051–1061, May 2009. [cited on p. 46]

T.L. Magnanti and R.T. Wong. Accelerating benders decomposition: Algorithmic enhancement and model selection criteria. *Oper. Res.*, 29(3):464–484, 1981. [cited on p. 31]

S. Martello, D. Pisinger, and P. Toth. Dynamic programming and strong bounds for the 0-1 knapsack problem. *Management Science*, 45(3):414–424, 1999. [cited on p. 84, 85, and 87]

S. Mattia. The robust network loading problem with dynamic routing. Talk in Aussois, 2010. [cited on p. 94]

R. Morris and D. Lin. Variance of aggregated web traffic. In *Proceedings of INFOCOM*, pages 360–366, 2000. [cited on p. 112]

D. P. Morton and R. K. Wood. *Advances in computational and stochastic optimization, logic programming and Heuristic Search*, chapter 5, pages 149–168. Woodruff, 1998. [cited on p. 75]

L. S. Moulin, M. Poss, and C. Sagastizábal. Transmission expansion planning with re-design. *Energy Systems*, 1(2):113–139, 2010. [cited on p. 46]

S. Mudchanatongsuk, F. Ordonez, and J. Liu. Robust solutions for network design under transportation cost and demand uncertainty. *J. Oper. Res. Soc.*, 59:552–562, 2008. [cited on p. 95]

J. M. Mulvey, R. J. Vanderbei, and A. Z. Stavros. Robust optimization of large-scale systems. *Oper. Res.*, 43(2):264–281, 1995. [cited on p. 76]

R. M. Nauss. Solving the generalized assignment problem: An optimizing and heuristic approach. *INFORMS J. on Computing*, 15(3):249–266, 2003. ISSN 1526-5528. [cited on p. 73]

G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, New York, 1999. [cited on p. 55 and 56]

J. P. Nolan. *Stable Distributions - Models for Heavy Tailed Data.* Birkhäuser, Boston, 2010. In progress, Chapter 1 online at academic2.american.edu/∼jpnolan. [cited on p. 108]

G. C. Oliveira, S. Binato, L. Bahiense, L. Thome, and M.V. Pereira. Security-constrained transmission planning: A mixed-integer disjunctive approach. In *Proc. IEEE/Power Eng. Soc. Transmission and Distribution Conf., São Paulo, Brazil*, 2004a. [cited on p. 58]

G. C. Oliveira, S. Binato, and M. V. F. Pereira. Value-based transmission expansion planning of hydrothermal systems under uncertainty. *IEEE TRANSACTIONS ON POWER SYSTEMS*, 622(4):1429–1435, 2007. [cited on p. 46, 68, and 116]

G.C. Oliveira, S. Binato, M.V.F. Pereira, and L.M. Thomé. Multi-stage transmission expansion planning considering multiple dispatches and contingency criterion. *Congresso Brasileiro de Automática*, sep. 2004b. [cited on p. 46]

S. Orlowski and R. Wessäly. An integer programming model for multi-layer network design. ZIB Preprint ZR-04-49, December 2004. [cited on p. 22]

S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly. SNDlib 1.0–Survivable Network Design Library. In *Proceedings of INOC, Spa, Belgium*, April 2007. http://sndlib.zib.de. [cited on p. 4, 20, and 36]

A. Ouorou and J.-P. Vial. A model for robust capacity planning for telecommunications networks under demand uncertainty. In *6th International Workshop on Design and Reliable Communication Networks, 2007. DRCN 2007*, pages 1–4. [cited on p. 95]

A. Renaud P. Tsamasphyrou and P. Carpentier. Transmission network planning: An efficient benders decomposition scheme. In *13th PSCC in Trondheim*, 1999. [cited on p. 68]

B. Pagnoncelli, S. Ahmed, and A. Shapiro. Sample average approximation method for chance constrained programming: Theory and applications. *Journal of Optimization Theory and Applications*, 142:399–416, 2009. [cited on p. 113]

S. Parikh. Lecture notes on stochastic programming. University of California, Berkeley, CA, 1968. [cited on p. 104]

K. Park, S. Kang, and S. Park. An integer programming approach to the bandwidth packing problem. *Manage. Sci.*, 42(9):1277–1291, 1996. ISSN 0025-1909. [cited on p. 111]

F. Pascali. *Chance Constrained Network Design.* PhD thesis, Universitá di Pisa, 2009. [cited on p. 112]

M. Patriksson. Simplicial decomposition algorithms. In *Encyclopedia of Optimization*, pages 3579–3585. 2009. [cited on p. 91]

M. Pereira and S. Granville. Analysis of the linearized power flow model in benders decomposition. Technical Report SOL 85-04, SOL Lab, Dept. of Oper. Res., Stanford University, 1985. [cited on p. 54]

M. Pióro and D. Medhi. *Routing, Flow, and Capacity Design in Communication and Computer Networks.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. [cited on p. 3, 19, and 96]

H. Pirkul and S. Soni. New formulations and solution procedures for the hop constrained network design problem. *Eur. J. Oper. Res.*, 148:126–140, 2003. [cited on p. 23]

D. Pisinger. An exact algorithm for large multiple knapsack problems. *European Journal of Operational Research*, 114(3):528 – 541, 1999. [cited on p. 89]

A. Prékopa. Dual method for a one-stage stochastic programming with random rhs obeying a discrete probability distribution. *Zeitschrift Oper. Res.*, 34:441–461, 1990. [cited on p. 104]

A. Prékopa. *Stochastic Programming.* Kluwer, 1995. [cited on p. 82, 104, 107, 108, and 117]

A. Prékopa. *Probabilistic programming models*, volume 10 of *In: Ruszczynski, A., Shapiro, A. (eds.) Stochastic Programming: Handbook in Operations Research and Management Science*, chapter 5, pages 267–351. Elsevier Science Ltd, Amsterdam, 2003. [cited on p. 103]

I. Quesada and I. E. Grossman. An LP/NLP based branch and bound algorithm for convex MINLP optimization problems. *Comput. Chem. Eng.*, 16(10/11):937–947, 1992. [cited on p. 53, 76, and 92]

C. Raack, A. M. C. A. Koster, and R. Wessaely. On cut-based inequalities for capacitated network design polyhedra. *Networks*, 2011. In press. [cited on p. 12 and 22]

R. Raman and I. E. Grossmann. Modeling and computational techniques for logic based integer programming. *Comput. Chem. Eng.*, 18(7):563–578, 1994. [cited on p. 54]

W. Rei, J.-F. Cordeau, M. Gendreau, and P. Soriano. Accelerating benders decomposition by local branching. *INFORMS J. Comput.*, 21(2):333–345, 2009. [cited on p. 32]

F.S. Reis, P.M.S. Carvalho, and L.A.F.M. Ferreira. Reinforcement scheduling convergence in power systems transmission planning. *Power Systems, IEEE Transactions on*, 20(2):1151–1157, May 2005. [cited on p. 46]

M. Riis and K. A. Andersen. Capacitated network design with uncertain demand. *INFORMS J. on Computing*, 14(3):247–260, 2002. ISSN 1526-5528. [cited on p. 94]

A. Saxena, V. Goyal, and M. A. Lejeune. MIP reformulations of the probabilistic set covering problem. *Math. Program.*, 121(1):1–31, 2009. [cited on p. 103 and 104]

M. G. Scutellà. On improving optimal oblivious routing. *Oper. Res. Lett.*, 37(3):197–200, 2009. [cited on p. 95]

S. Sen, R. D. Doverspike, and S. Cosares. Network planning with random demand. *Telecommunication Systems*, 3(1):11–30, 1994. [cited on p. 75, 94, and 112]

A. Shapiro, D. Dentcheva, and A. Ruszczynski. *Lectures on Stochastic Programming: Modeling and Theory.* SIAM, Philadelphia, 2009. [cited on p. 103]

Id.J. Silva, M.J. Rider, R. Romero, and C.A.F. Murari. Transmission network expansion planning considering uncertainty in demand. *Power Systems, IEEE Transactions on*, 21(4):1565–1573, Nov. 2006. [cited on p. 46]

K. Singh, A. Philpott, and K. Wood. Column-generation for design of survivable networks. Working paper, 2008. [cited on p. 51]

D. Spoerl and R.K Wood. A stochastic generalized assignment problem. INFORMS Annual Meeting, Atlanta, GA, 19–22 October, 2003. [cited on p. 109]

O.B. Tor, A.N. Guven, and M. Shahidehpour. Congestion-driven transmission planning considering the impact of generator expansion. *Power Systems, IEEE Transactions on*, 23(2):781–789, May 2008. [cited on p. 46]

P. Tsamasphyrou, A. Renaud, and P. Carpentier. Transmission network planning under uncertainty with benders decomposition. *Lecture Notes Econ. Math. Systems*, 481:457–472, 2000. [cited on p. 31 and 46]

R. Uehara Y. Uno. On computing longest paths in small graph classes. *International Journal of Foundations of Computer Science*, 18(5):911–930, 2007. [cited on p. 64]

C. van de Panne and W. Popp. Minimum-cost cattle feed under probabilistic protein constraints. *Management Science*, 9(3):405–430, 1963. [cited on p. 104]

R. Villanasa. *Transmission network planning using linear and mixed linear integer programming.* PhD thesis, Ressenlaer Polythechnic Institute, 1984. [cited on p. 54]

L.A. Wolsey. *Integer Programming.* Wiley, 1998. [cited on p. 8]

R. T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical Programming*, 28(3):271–287, 1984. [cited on p. 53]

K. A. Woolston and S. L. Albin. The design of centralized networks with reliability and availability constraints. *Comput. & OR*, 15(3):207–217, 1988. [cited on p. 20]

D. Yuan. *An annotated bibliography in communication network design and routing.* PhD thesis, Institute of Technology, Linköpings Universitet, 2001. [cited on p. 11 and 22]

J. H. Zhao, Z. Y. Dong, P. Lindsay, and K. P. Wong. Flexible transmission expansion planning with uncertainties in an electricity market. *Power Systems, IEEE Transactions on*, 24(1):479–488, Feb. 2009. [cited on p. 46]