Dual approaches in stochastic programming

Marc Letournel

UNIVERSITE PARIS-SUD
ÉCOLE DOCTORALE : Laboratoire de Recherche en Informatique.
DISCIPLINE Graphes Combinatoires.

THÈSE DE DOCTORAT

soutenue le 27/09/2013

par

Marc LETOURNEL

APPROCHES DUALES DANS LA RESOLUTION DE PROBLEMES STOCHASTIQUES

Directeur de thèse : Abdel LISSER Professeur, Laboratoire de Recherche en Informatique, Orsay.

Président du jury : Marc BABOULIN, Professeur Université Paris Sud.

Rapporteurs :

Jean-Baptiste HIRIART-URRUTY, Professeur, Université de Toulouse.

Alexei GAIVORONSKI, Professeur, Université de Trondheim (Norvège).

Examinateur :

Patrice PERNY, Professeur Université Paris 6.

A ma femme, sans qui je ne suis rien,

à ma mère, parce qu'elle me manque,

à mes enfants, car l'avenir leur appartient.

Remerciements

Les choix qui gouvernent notre vie sont faits de petits instants, de rencontres improbables, de décisions simples, qui nous dirigent à un instant donné dans une direction. En revanche, ces orientations deviennent fondamentales, et nous y engageons tous nos efforts, parce que des hommes et des femmes nous soutiennent, nous aident, et s'impliquent à nos côtés dans nos entreprises. C'est presque par hasard que j'ai entrepris ce travail de thèse. Lorsque j'ai poussé la porte du LRI, il y a quelques années, mon intention était de suivre quelques cours d'informatique pour remettre à niveau mes connaissances dans ce domaine, après quinze années d'enseignement des mathématiques en classes préparatoires. Mon idée était alors simplement de suivre quelques modules pour rafraîchir mes connaissances en réseau ou en programmation. C'est Daniel Etiemble qui m'a tout naturellement suggéré de m'inscrire complètement au master recherche du LRI, c'est ensuite Emmanuel Waller qui m'a proposé de passer les examens, puisque j'étais inscrit officiellement. Et c'est enfin Abdel Lisser qui m'a proposé de continuer l'aventure à l'issue de son cours sur la recherche opérationnelle, tout d'abord par un mémoire sur la programmation semi définie, puis par une thèse sur les problèmes d'intégralité dans les problèmes stochastiques combinatoires. Evidemment, le professeur Lisser est la première personne à qui j'adresse mes remerciements, il a su accompagner mes efforts tout en s'adaptant à ma situation particulière de thésard de plus de quarante ans, exerçant une activité professionnelle à plein temps. Il est certain que ses directives ont été cruciales pour le bon déroulement de cette thèse.

Une de mes motivations pour cette thèse, a été la possibilité qu'elle m'offrait de travailler avec d'autres personnes. Le métier d'enseignant, en dehors de phases administratives, reste essentiellement un métier qu'on exerce seul. Au laboratoire de recherche, j'ai pu collaborer, échanger, participer avec d'autres chercheurs sur le coeur de leur activité. Je remercie tous ceux et celles qui m'ont accueilli, donné de leur temps, échangé leur collaboration. Merci à Charles, Pablo, Rafael, Pierre, Stéfanie, Odile, Paul Dorbec, Cheng, et d'autres encore de l'équipe GraphComb. Je remercie également le professeur Rüdiger, pour les après-midi passés à dessiner des graphes multi-scénarios, ou à réfléchir ensemble sur des matrices de 0 *et de* 1*. Je remercie également mes rapporteurs de thèse, les professeurs Hiriart-Urruty et Gaivoronski, pour avoir accepté cette tâche, et m'avoir communiqué leur questions et corrections pendant cette dernière phase de soumission. Je remercie aussi les*

membres du jury, les professeurs Patrice Perny et Marc Baboulin, pour le temps qu'il consacrent à mon travail.
Bien évidemment, j'accorde une place toute particulière à ma famille dans mes remerciements. Tous ceux qui me connaissent savent combien elle compte pour moi. Merci à mon épouse Anne-Catherine, pour tous ces instants où tu partages mes efforts, mes doutes et mes joies. Merci à mes enfants pour supporter un père qui travaille même quand il n'est pas à son travail. J'ai une ultime pensée pour ma mère qui vient de nous quitter à quatre mois de l'échéance, elle qui n'a jamais douté de ma capacité à aller au bout de mes efforts.

Contents

Introduction

Le travail général de cette thèse consiste à étendre les outils analytiques et algébriques usuellement employés dans la résolution de problèmes combinatoires algorithmiques déterministes à un cadre combinatoire stochastique. Deux cadres distincts sont abordés : les problèmes combinatoires stochastiques discrets et les problèmes stochastiques continus. Le cadre discret est abordé à travers le problème de la forêt couvrante de poids maximal dans une formulation Two-Stage à multi scénarios. La version déterministe très connue de ce problème établit des liens entre la fonction de rang dans un matroïde et la formulation duale via l'algorithme glouton. La clé de voûte de la preuve mathématique du cas déterministe réside d'une part dans la fomulation duale du problème et l'absence de saut de dualité pour le problème linéaire, et d'autre part dans une transformation d'Abel appliquée sur la différence de coût des arêtes. La formulation stochastique discrète du problème de la forêt maximale couvrante est transformée en un problème déterministe équivalent, mais du fait de la multiplicité des scénarios, le dual associé est en quelque sorte incomplet. Le travail réalisé ici consiste à comprendre en quelles circonstances la fomulation duale atteint néanmoins un minimum égal au problème primal intégral. D'ordinaire, une approche combinatoire classique des problèmes de matroïdes consiste à rechercher des configurations particulières au sein des graphes, comme les circuits, et à explorer d'éventuelles recombinaisons. Le problème classique de l'intersection de deux matroïdes est par exemple résolu par ce type d'approche algorithmique, où la partie analytique est finalement absente. Les preuves combinatoires prennent en compte les éléments de reconfiguration d'un graphe pondéré en inventoriant une liste de reconfigurations possibles . Pour donner une interprétation prosaïque, si on change d'une manière infinitésimale les valeurs de poids des arêtes d'un graphe, il est possible que la forêt couvrante se réorganise complètement. Ceci est vu comme un obstacle dans une approche purement combinatoire. Pourtant, certaines grandeurs analytiques vont varier de manière continue en fonction de ces variations infinitésimales, comme la somme des poids des arêtes choisies. Il apparaît également que les choix de telle ou telle arête est une fonction de son poids, mais également du poids des autres arêtes. Ainsi, il est naturel d'essayer de formuler ces sauts décisionels comme autant de fonctions implicites (je serais tenté d'écrire fonctions implicites les unes des autres si cela n'était par essence même le rôle des fonctions implicites). Après un premier chapitre d'introduction

des concepts de base, la section 2 décrit la formulation déterministe et la formulation stochastique du problème de la forêt couvrante de poids maximal. Signalons dès à présent que j'ai choisi de répartir les références bibliographiques dans chacun des chapitres séparément, dans la mesure où la lecture des chapitres peut se faire séparément elle aussi. Dans le chapitre 3, la formulation stochastique de la forêt couvrante dans le cas de deux scénarios seulement est abordée avec une preuve de la conservation du caractère intégral du dual. Le chapitre 4 présente le cas de trois scénarios ou plus et donne les situations où le système dual perd son caractère intégral. Le chapitre 5 propose une réduction du problème considéré et aborde un algorithme d'approximation dans le cas d'un dual non intégral. Dans le cas où le dual n'est pas intégral, on peut explorer les forêts couvrantes après relaxation du problème. Une autre difficulté surgit liée au fait que le nombre d'inégalités du système est exponentiel. En effet, pour chaque sous ensemble de sommets, une contrainte apparait dans le fait que le nombre d'arêtes internes doit être strictement inférieur au cardinal de l'ensemble de sommets. Le chapitre 6 propose un modèle polynomial de contraintes par rapport au cardinal de l'ensemble de sommets en introduisant une orientation arbitraire des arêtes, des résultats numériques sont présentés dans une mise en oeuvre du modèle. Les problèmes stochastiques continus sont abordés au cours du chapitre 7 dans le cadre du problème de sac à dos avec contrainte stochastique. La formulation est de type "chance constraint", et la dualisation par variable lagrangienne est adaptée à une situation où la probabilité de respecter la contrainte doit rester proche de 1. Le modèle étudié est celui d'un sac à dos où les objets ont une valeur et un poids déterminés par des distributions normales. Cette situation présente un certain nombre d'avantages calculatoires. En premier lieu, la contrainte étant linéaire, son expression devient une espérance d'une loi normale. Cette formulation permet de s'affranchir de problèmes de convexité, voire de connexité de l'espace admissible des solutions. De plus, la loi normale étant déterminée par sa moyenne et son écart-type, il est possible de géométriser complètement le problème. C'est cette particularité qui est exploitée par Prékopa dans [50] pour affirmer que le problème est convexe pour $p > \frac{1}{2}$, mais c'est également la même particularité qui permet de mettre en oeuvre une résolution par la méthode du "second order cone programming". Dans notre approche, nous nous attachons à appliquer des méthodes de gradient directement sur la formulation en espérance de la fonction objectif et de la contrainte. Nous délaissons donc une possible reformulation du problème sous forme géométrique pour détailler les conditions de convergence de la méthode du gradient stochastique. Cette partie est illustrée par des tests numériques de comparaison avec la méthode SOCP sur des instances combinatoires avec méthode de Branch and Bound, et sur des instances relaxées.

The global purpose of this thesis is to study the conditions to extend analytical and algebraical properties commonly observed in the resolution of determin-

istic combinatorial problems to the corresponding stochastic formulations of these problems. Two distinct situations are treated : discrete combinatorial stochastic problems and continuous stochastic problems. Discrete situation is examined with the Two Stage formulation of the Maximum Weight Covering Forest. The well known corresponding deterministic formulation shows connexion between the rank function of a matroid, the greedy algorithm , and the dual formulation. The key-stone of the mathematical proof in the deterministic case relies on the dual formulation and on an Abel transformation on the summation of the costs of the greedy selected edges. The discrete stochastic formulation of the Maximal Covering Forest is turned into a deterministic equivalent formulation, but, due to the number of scenarios, the associated dual is not complete. The work of this thesis leads to understand in which cases the dual formulation of the primal problem remains integer, and, when it is not the case, how to approximate the primal integer solution. Usually, classical combinatorial approaches aim to find particular configurations in the graph, as circuits, in order to handle eventual reconfigurations. The classical problem of intersection of two matroids is solved by this kind of methods, where we can consider that analytical considerations are not taken in account. Combinatorial proofs examine possible reconfiguration of a weighted graph by listing all combinations of specific edges. In order to give a global consideration, slight modifications of the weights of the edges mights change considerably the configuration of the Maximum Weight Covering Forest. This can be seen as an obstacle to handle pure combinatorial proofs. However, some global relevant quantities, like the global weight of the selected edges during the greedy algorithm, have a continuous variation in function of the weights of each edges in the graph. It appears equally, that an edge is selected or not depending on its own weight, but equally depending on the weight of the other edges too. So it is attracting to try to formulate this decision as an implicit function. After a first chapter devoted to the introduction of basis concepts, like matroids, rank function, duality, System Totally Dual Integral, the chapter 2 describes the deterministic formulation of the Maximum Weight Covering Forest. I Have chosen to give bibliographical references separately in each chapter, in consideration to the relative main distance between the different subjects in the thesis. In chapter 3, we deal with the stochastic formulation of the covering forest in the case of only two scenarios, and we give the proof of the TDIness of the problem. Chapter 4 is devoted to the case of more or equal to three scenarios and give conditions to preserve TDIness. Chapter 5 gives a reduction of this problem and elaborates an approximation algorithm in the case where the system is not TDI. In the case where the dual formulation is not integer, we come back to a direct resolution of the primal problem. But another difficulty is that the number of constraints grows exponentially with the number of vertices. Every subset of vertices is associated to a new constraint : the number of chosen edges connecting these vertices must be strictly smaller than the number of vertices itself. Chapter 6 proposes a polynomial formulation of the constraints by introducing an arbitrary orientation of the edges. Numerical experiments are presented.

Continuous stochastic problems are presented in chapter 7 in the case of the stochastic Knapsack with chance constraint. Chance constraint and dual Lagrangian formulation are adapted in the case where the expected probability of not exceeding the knapsack capacity is close to 1. The model introduced consists in item with costs and rewards following a normal distribution. This situation is comfortable in the sense that in some cases, the convexity and even the connectedness of the feasible set is not guaranteed for some stochastic process. In case of normal distribution, completely determined by its mean and its standard deviation, the feasible set gains a geometric description that ensures easy computations, with a new formulation of the constraint (SOCP method). In our case, we try to apply direct gradient methods rather than reformulating the problem in geometrical terms. We detail convergence conditions of gradient based methods directly on the initial formulation. This part is illustrated with numerical tests on combinatorial instances and Branch and Bound evaluations on relaxed formulations.

Part I

Discrete Stochastic modeling

Chapter 1

Matroids, sub-modular functions, duality

1.1 Introduction

This chapter introduces the fundamental tools used to modelize the discrete problem of finding a Maximal Weight Covering Forest in a graph. These tools are Matroids, rank function, dual formulation, and System Totally Dual Integral. The purpose of this part is not to present a complete and deep overview of these subjects, but to pick up progressively the different aspects which are relevant for the proofs we will give in the different parts of our work. Matroid theory is developed in [49]. Matroids have been introduced by Whitney(1935) to study the fundamental properties of dependence that appears commonly in a graph and in a matrix. The first chapter is organised as follow : in section 1.2, we introduce several equivalent definitions of matroids, in section 1.3, we introduce the rank function and the notion of submodular function, in section 1.4.1, we present a first approach of dual formulation for a linear problem, and finally, in section 1.4.3, we introduce the notion of System Totally Dual Integral.

1.2 General definitions

Several equivalent definitions of matroids exist. We use the following approach given in [48]:

Definition 1 *Let $N = \{1, \ldots, n\}$ be a finite set, and consider a collection $\mathscr{F}$ of subsets. $(N, \mathscr{F})$ is an independence system if*

$$\forall F_1 \in \mathscr{F}, \forall F_2 \subset N, F_2 \subset F_1 \Rightarrow F_2 \in \mathscr{F}.$$

Elements of $\mathscr{F}$ are called independent sets, and the remaining subsets are called dependent sets.

Definition 2 *Given an independence system $(N, \mathscr{F})$, a subset $F \in \mathscr{F}$ is said to be a maximal independent set if $F \cup \{j\} \notin \mathscr{F}$ for all $j \notin F$.*

Definition 3 *A maximal independent set T is maximum if $|S| \leq |T|$ for all $S \in \mathscr{F}$.*

We introduce the notation $m(T) = \{\max_{S \subset T} |S| : S \in \mathscr{F}\}$.

Definition 4 *$M = (N, \mathscr{F})$ is a matroid if M is an independence system in which for any subset $T \subset N$, every independent set in T that is maximal in T has the same cardinality $m(T)$.*

Some other equivalent definitions can be adopted, using a "switching axiom" formulation:

Definition 5 *Alternative formulation: Let $N = \{1, \ldots, n\}$ be a finite set, and consider a collection $\mathscr{F}$ of subsets. $(N, \mathscr{F})$ is a matroid if*

- $\emptyset \in \mathscr{F}$
- $\forall F_1 \in \mathscr{F}, \forall F_2 \subset N, F_2 \subset F_1 \Rightarrow F_2 \in \mathscr{F}$
- $\forall F_1 \in \mathscr{F}, \forall F_2 \in \mathscr{F}, |F_2| < |F_1| \Rightarrow \exists j \in F_1 \backslash F_2$ *such as* $j \cup F_2 \in \mathscr{F}$.

Among independent sets, those with maximum cardinality are called bases:

Definition 6 *Let $M = (N, \mathscr{F})$ be a matroid, a set B is a base if $B \in \mathscr{F}$ and $\forall j \in N, j \cup B \notin \mathscr{F}$*

It is possible to define matroids from the collection of bases:

Definition 7 *Let N be a finite set and $\mathscr{B}$ a collection of subsets such that:*

- $\mathscr{B} \neq \emptyset$
- $\forall (B_1, B_2) \in \mathscr{B}$ *and* $i \in B_1 \backslash B_2, \exists j \in B_2 \backslash B_1$ *such that* $(B_1 - \{i\}) \cup \{j\} \in \mathscr{B}$

Independent sets are subsets of elements of $B \in \mathscr{B}$.

Circuits are subsets with an extra element:

Definition 8 *Circuits. Let M be a matroid. A subset $C \subset N$ is a circuit if*

- $C \notin \mathscr{F}$
- $\forall j \in C, C \backslash j \in \mathscr{F}$

Circuit is synonym of cycle for graphic matroids. It is possible to define matroids from the collection of circuits:

Definition 9 *Let N be a finite set and $\mathscr{C}$ a collection of subsets such that:*

- $\emptyset \notin \mathscr{C}$
- $\forall (C_1, C_2) \in \mathscr{C}^2, C_1 \subseteq C_2 \Rightarrow C_1 = C_2$
- $\forall (C_1, C_2) \in \mathscr{C}^2$ *and* $C_1 \neq C_2$, $\forall i \in C_1 \cap C_2, \exists C_3 \in \mathscr{C}, C_3 \subset (C_1 \cup C_2) \backslash \{i\}$

For the problem of finding a maximal independent covering forest in a graph, these formulations are useful to construct algorithms where the global strategy is based on exchanging the edges one by one. But, according to the approach we lead in this work, considering the first definition is more adapted. We do not compare the independent sets with the analysis of closed configurations between themselves, but we compare the independent sets by ranking the weight of the lightest edge of the subset. This way is more flexible and it is possible to handle simultaneously several subsets with different configurations but with a common level of weighted edges.

There exist several kinds of matroids. First examples have been introduced in 1935 by Whitney in matrices when considering the subsets of columns who are linearly independent.

1.2.1 Matric and linear matroids

Matric matroid

For any matrix $A \in \mathscr{M}_{m,n}(\mathbb{K})$, consider the set N of columns of A noted $\{C_1, \ldots, C_n\}$. The family $\mathscr{F}$ of independent subsets consists of subset of columns who are linearly independent. $m(T)$ is the space dimension of the subspace generated by a family T of columns.

Linear matroid

The pending formulation of a matric matroid in terms of vectors of $\mathbb{K}^m$ is given by the independent vectors of a family of N vectors corresponding to the columns of a matrix $A \in \mathscr{M}_{mn}(\mathbb{K})$.

1.2.2 Graphic matroids

This type of matroids is widely used in this work. Let $G = (V, E)$ be a graph, and $\mathscr{F}$ the collection of subsets whose edges contain no cycles. $M = (E, \mathscr{F})$ is a matroid.

- The independent sets are the forests.
- The circuits are the cycles of G.
- The bases are the maximal covering forests.

Directed graphs can be used in the same manner: Let $G = (V, E)$ be a digraph, and $\mathscr{F}$ the collection of subsets whose edges contain no cycles. $M = (E, \mathscr{F})$ is a matroid.
A global survey of Franck [21] for applications of submodular functions presents properties of directed graphs and associated matroids.

Isomorphism and matroids

Definition 10 *Two matroids $M = (N, \mathscr{F})$ and $M' = (N', \mathscr{F}')$ are isomorphic if there exists a bijection f that maps independent sets of M to independent sets of M'.*

Theorem 11 *Any graphic matroid is isomorphic to a matric matroid.*

Proof. This theorem is given in [49]. Let $G = (V, E)$ be a graph, and for any vertex $i \in V$, consider the vector

$$V_i = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

where 1 is on the line i. For any edge $(u, v) \in E$, build the vector $V_{uv} = V_u - V_v$. For any circuit $C = \{u_1u_2, u_2u_3, \ldots, u_ku_1\}$ in the graph, the corresponding family $S_C = \{V_{u_1u2}, \ldots, V_{u_ku_1}\}$ is linearly dependent. Moreover for any egde in the circuit, let say (u_ku_1), the family $S_C \backslash \{V_{u_ku_1}\}$ is linearly independent.

Conversely, it is much more difficult to associate any matric matroid with a graphic matroid. If we consider a similar reverse construction with any matrix $A \in \mathscr{M}_{mn}(\mathbb{K})$ of rank r, we try to construct a graph in this way : We call $\{(C_1, \ldots, C_n)\}$ the columns of A. There exists a subset of columns of cardinal r which is a base, call it B and suppose without restriction that B consists of the r first columns set of A. For any columns C_i with $i > r$ there exists only one subset of B (and moreover only one linear combination) of elements of B such that C_i belongs to the subspace generated by this subset.

$$\forall i > r, \exists! \{j_{1i}, j_{2i}, \ldots, j_{ki}\} \subset \{1, \ldots, r\}, C_i \in Vect(C_{j_{1i}}, \ldots, C_{j_{ki}})$$

The indexes $\{j_{1i}, j_{2i}, \ldots, j_{ki}\}$ are considered in an increasing order. Consider the map ϕ that associates to a column C_i the following column of a new matrix A':

$$\forall i \in \{1, \ldots, r\}, C_i \mapsto V_{i,i+1} = \begin{pmatrix} 0 \\ \vdots \\ 1 \\ -1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

where 1 is on the line i. And

$$\forall i \in \{r+1, \ldots, n\}, C_i \mapsto V_{j_{ki} j_{1i}} = \begin{pmatrix} 0 \\ \vdots \\ -1 \\ 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

where 1 is on the line j_{ki} and -1 on the line j_{1i}. Now we associate to A' a graphic matroid when considering a graph of $r+1$ vertices, where the edge between two vertices exist if and only if the corresponding column exist in the matrix A'. Unfortunately, this map is not a bijection. Suppose that there exist two different columns C and C' that are exactly in the same subspace generated by $Vect(C_{j_{1i}}, \ldots, C_{j_{ki}})$. These two columns are associated to the same edge.
Yet, it is possible to associate a matric matroid to a graphic matroid with an exponential number of edges, and make a bijection between circuits. We show an example with a matrix $A \in \mathscr{M}_{n4}(\mathbb{K})$ whose rank is 2 in figure 1.1. Suppose that any subset of two columns is linearly independent (for instance $A = (C_1 C_2 C_3 C_4)$ is the matrix associated with 4 non collinear vectors in a plan of $\mathbb{K}^n$). Circuits of A are all the subsets of 3 vectors.
The map between the set of columns of A and the subsets of G is the following one:

$$\begin{array}{c} \{C_1\} \to \{e_{11}, e_{12}, e_{13}\} \\ \{C_2\} \to \{e_{21}, e_{22}, e_{24}\} \\ \{C_3\} \to \{e_{31}, e_{33}, e_{34}\} \\ \{C_4\} \to \{e_{42}, e_{43}, e_{44}\} \end{array}$$

Note that the graph G is not connected. Indeed, the bijection is not directly between the edges of the graph and the columns of the matrix.

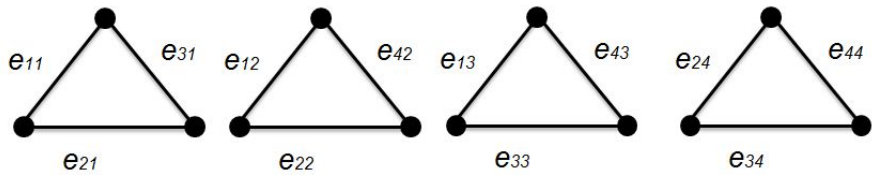

Figure 1.1: graphic matroid associated with A

■

Indeed, the question of representation of any matroid by a matric matroid has huge developments. First of all, in this short introduction, we just consider matrix with $0, 1$ coefficients. Considering matrices with coefficients in a finite field modify linear dependence other the columns of a matrix, and open widely the possible matric reprensentation for a matroid. It is well known that any graphic matroid is isomorphic with a matric matroid over any finite field [49]. Matroids that admit a graphic or matric representation are said to have a compact representation.

1.2.3 Partition matroid

Given m disjoint finite sets E_i for $i \in \{1, \ldots, m\}$, let $E = \bigcup_{i=1}^{m} E_i$, $F \subset E$ is independent if $|F \cap E_i| \leq 1, \forall i \in \{1, \ldots, m\}$. $(E, \mathscr{F})$ is a matroid.

Partition matroids can be used to show that the *k-matroid intersection problem* is NP Complete with $k \geq 3$ [48]: Given $k \geq 3$ matroids $M_i = (N, \mathscr{F}_i)$ for $i \in \{1, \ldots, k\}$ and a weight vector $c \in \mathbb{R}^n$, the problem is

$$\max_S \left\{ \sum_{j \in S} c_j : S \in \bigcap_{i=1}^{k} \mathscr{F}_i \right\}.$$

The intersection problem of 3 matroids can be reduced to the search of an Hamiltonian path in a graph. Consider a directed graph $\mathscr{D} = (V, \mathscr{A})$ and the following matroids:

- $M_1 = (\mathscr{A}, \mathscr{F}_1)$ where $\mathscr{F}_1$ are the subsets of $\mathscr{A}$ with no cycle.
- $M_2 = (\mathscr{A}, \mathscr{F}_2)$ where $\mathscr{F}_2$ are the subsets of $\mathscr{A}$ where, for each vertex e of V, there is at most only one arc entering on e.
- $M_3 = (\mathscr{A}, \mathscr{F}_3)$ where $\mathscr{F}_3$ are the subsets of $\mathscr{A}$ where, for each vertex e of V, there is at most only one arc leaving e.

M_1 is a graphic matroid and M_2 and M_3 are partition matroids. Every subset in the intersection $M_1 \cap M_2 \cap M_3$ is an Hamiltonian path.

There exist several class of matroids which are not presented here. We focus here on graphic matroids since we study the maximum weight forest problem in a graph. Graphic matroids are said to be *binary* matroids because they can be represented by a matrix with coefficients in the field $\mathbb{K}$ with two elements $\{0, 1\}$.

1.3 Rank function

Definition 12 *Consider a matroid $M = (V, \mathscr{F})$, the cardinality function $m(T) = \max_{F \subset T} \{|F|, F \in \mathscr{F}\}$ is the rank function associated to the matroid M.*

Definition 13

- *A function f is submodular on V if*

$$\forall (S, T) \subset V^2, f(T \cap S) + f(T \cup S) \leq f(S) + f(T)$$

- *A function f is nondecreasing on V if*

$$\forall (S, T) \subset V^2, T \subset S \Rightarrow f(T) \leq f(S)$$

The rank function m associated to a matroid is submodular and nondecreasing.

In the same way that matroids can be defined by different approaches (bases, independent sets, cycles,...), a rank function can be the starting point to define matroids [48]:

Proposition 14 *An independence system $(V, \mathscr{F})$ whose cardinality function m is submodular is a matroid.*

A more generaly approach can be formulated in this way:

Definition 15 *A function $f : \mathcal{P}(V) \to \mathbb{N}$ such that*

1. $f(\emptyset) = 0$
2. $\forall e \in V, f(e) = 1$
3. $\forall F \in \mathcal{P}(V), \forall e \in V, f(e \cup F) = f(F) + k(F, e)$ *where* $k(F, e) \in \{0, 1\}$
4. $\forall F \in \mathcal{P}(V), \forall (e_1, e_2) \in V^2, f(e_1 \cup F) = f(e_2 \cup F) = f(F) \Rightarrow f(F \cup e_1 \cup e_2) = f(F)$.

is called a rank function.

From this starting point, it is possible to define a matroid in a set by its bases: subsets $F \neq \emptyset$ such that $f(F) = f(V)$ and $\forall e \in F, f(V \backslash \{e\}) < f(F)$.

Before ending this introduction on matroids, it is essential to introduce the concept of closure.

Definition 16 *Let $M = (V, \mathscr{F})$ be a matroid, with a rank function m. The closure of a subset F is $cl(F) = \{e \in V, m(F \cup \{e\}) = m(F)$. The closure of a set F is called equally the span of F.*

This definition is central in the first part of this work. The closure is clearly connected to the notion of cycle presented in definition 8. Moreover, there is another way to define a matroid, which is not so much used in graphs. This definition was given as the first step of a former course of mathematics on combinatorial geometry [20] and we will see that this definition useful in our work :

Definition 17 *A dependency closure on V is an application $cl : \mathcal{P}(V) \mapsto \mathcal{P}(V)$ which satisfies:*

1. $\forall F \subset V, F \subset cl(F)$
2. $\forall (F_1, F_2) \subset V^2, F_1 \subset F_2 \Rightarrow cl(F_1) \subset cl(F_2)$
3. $\forall F \subset V, cl(F) = cl(cl(F))$
4. $\forall F \subset V, \forall (x, y) \in V^2, (y \in cl(F \cup \{x\})$ and $y \notin cl(F)) \Rightarrow x \in cl(F \cup \{y\})$
5. $\forall F \subset V, \forall x \in cl(F), \exists Z \subset F, |Z|$ finite and $x \in cl(Z)$

The last axiom is clearly useless in finite sets, but ordinary expresses that the algebraic dependency is over a finite subset.

Then, the couple $M = (V, cl)$ is a matroid. It is easy to define a base as a subset of V with minimal cardinality and whose closure is exactly V.

1.4 Duality, System totally Dual Integral

1.4.1 Duality

In mathematics and specially in graph theory, duality can be traduced in numerous ways. Finding a duality formulation to a problem means generally that we transport the problem into another space configuration with an isomorphism, and that the new structure is easier to handle. Consider the following maximization problem : $c \in \mathbb{R}^n$, $A \in \mathscr{M}_{pn}(\mathbb{R})$, $x \in \mathbb{R}^n$ and $b \in (\mathbb{R}^+)^p$

$$\mathrm{Z}_{LP} = \begin{cases} max \sum_{j=1}^{n} c_j x_j \\ Ax \leq b \end{cases} \quad (1.1)$$

We consider the columns of the matrix A as vectors $V_1, \ldots, V_n$ of $\mathbb{R}^p$. The first interpretation of 1.4.1 is to attribute a specific weight c_j to each vector V_j and to find the maximum weighted linear combination of $\{V_1, \ldots, V_n\}$ in the part P of the associated half cone bounded by b:

$$P = \{(x_1, \ldots, x_n) \in (\mathbb{R}^+)^n \backslash x_1 V_1 + \ldots + x_n V_n \leq b\}$$

P is a polyhedron and we outline that a topological argument shows that Z_{LP} has always a solution which is reached whenever $A \in \mathscr{M}_{pn}(\mathbb{R}^+)$ and $b \in Vect(V_1, \ldots, V_n)$ since P is a nonempty compact subset of $\mathbb{R}^n$ (closed and bounded) while $\sum_{j=1}^{n} c_j x_j$ is a continuous function of $(x_1, \ldots, x_n)$.

A dual approach leads to considering each line $L_1, \ldots, L_p$ of the matrix A as a linear form on the space $\mathbb{R}^n$, and the combination $\sum_{j=1}^{n} c_j x_j$ as a linear form $\phi(x_1, \ldots, x_n)$. If we note $L_i(x_1, \ldots, x_n) = \sum_{j=1}^{n} A_{ij} x_j$, and according to the fact that $\phi \in Vect(L_1, \ldots, L_p)$, Z_{LP} can be interpreted as the maximization of ϕ over $(\mathbb{R}^+)^n$ under the constraints that each linear form L_i is bounded by b_i over the same subpart of the space. The dual approach leads to the dual formulation that we introduce with an example in the next subsection:

1.4.2 Dual formulation

Consider the following problem:

$$Z_{LP} = \begin{cases} \max 2x_1 + x_2 + 2x_3 \\ x_1 + x_2 + x_3 \leq 2 \\ x_2 \leq 1 \\ x_1 + x_2 \leq 1 \end{cases} \quad (1.2)$$

The first approach is to maximize the linear combination of $V_1 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$, $V_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ and $V_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$ bounded by $b = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}$ according to the fact that the costs of V_1 and V_3 are 2 while the cost of V_2 is only 1. In our example, $P = \{(x_1, x_2, x_3) \in (\mathbb{R}^+)^3 \backslash x_1 V_1 + x_2 V_2 + x_3 V_3 \leq b\}$

The dual approach leads to maximize the linear form $\phi(x_1, x_2, x_3) = 2x_1 + x_2 + 2x_3$ under the constraints that the three linear forms L_1, L_2, L_3 are bounded respectively by : $L_1(x_1, x_2, x_3) = x_1 + x_2 + x_3 \leq 2$, $L_2(x_1, x_2, x_3) = x_2 \leq 1$ and $L_3(x_1, x_2, x_3) = x_1 + x_2 \leq 1$.
In this case, observe that A is invertible, so that ϕ is a linear combination of L_1, L_2, L_3. The combination is unique : $\phi = 2L_1 - L_2$.
Now we introduce $(y_1, y_2, y_3) \in (\mathbb{R}^+)^3$ and consider $y_1L_1 + y_2L_2 + y_3L_3$. Since (L_1, L_2, L_3) is a basis of the dual space of $\mathbb{R}^3$, it is possible to find (y_1, y_2, y_3) such that $\phi \leq y_1L_1 + y_2L_2 + y_3L_3$ (it suffices to increase the linear combination $\phi = 2L_1 - L_2$ up to positive coefficients)[1]. For any vector $x \in (\mathbb{R}^+)^3$ and $y \in (\mathbb{R}^+)^3$, we get:

$$\left\{ \begin{array}{c} \phi \leq y_1L_1 + y_2L_2 + y_3L_3 : (y_1, y_2, y_3) \in (\mathbb{R}^+)^3 \\ \Rightarrow \phi(x) \leq y_1L_1(x) + y_2L_2(x) + y_3L_3(x) : x \in (\mathbb{R}^+)^3 \\ x \in P \Rightarrow \phi(x) \leq y_1L_1(x) + y_2L_2(x) + y_3L_3(x) \leq y_1b_1 + y_2b_2 + y_3b_3 \end{array} \right.$$

If we introduce the dual problem Z_{LD}

$$Z_{LD} = \left\{ \begin{array}{c} \min \sum_{i=1}^{3} y_ib_i \\ \phi \leq y_1L_1 + y_2L_2 + y_3L_3 \\ (y_1, y_2, y_3) \in (\mathbb{R}^+)^3 \end{array} \right.$$

then :

$$\max_{x \in P} \phi(x) \leq \max_{\substack{x \in P \\ y \notin P'}} y_1L_1(x) + y_2L_2(x) + y_3L_3(x) \leq y_1b_1 + y_2b_2 + y_3b_3 : y \notin P'$$

Note that the matrix formulation of the constraint of Z_{LD} is $A^Ty \geq c$ and a dual polyhedron can be defined by : $P' = \{(y_1, y_2, y_3) \in (\mathbb{R}^+)^3 \backslash A^Ty < c\}$. The right hand side of this equation does not depend on x, and it results that $Z_{LP} \leq Z_{LD}$. The difference $Z_{LD} - Z_{LP}$ is called the duality gap. This example illustrates the weak LP duality theorem.

Indeed, this gap is zero, this result is the strong duality theorem. We keep the same example to illustrate the mechanism of the strong duality theorem.
We have seen that (L_1, L_2, L_3) is a basis of the linear forms on $(\mathbb{R}^+)^3$. We can compute the ante-dual basis $(\epsilon_1, \epsilon_2, \epsilon_3)$, verifying $L_i(\epsilon_i) = \delta_{ij}$ for $(i, j) \in [1, 3]^2$.
Obviously, the column vectors $(\epsilon_1, \epsilon_2, \epsilon_3)$ are the columns of the matrix $A^{-1} =$

[1] More generally, we need that $\phi \in Vect(L_1, \ldots, L_p)$, and a condition that ensures that is that $\{L_1, \ldots, L_p\}$ is a generating system

$$\begin{pmatrix} 0 & -1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & -1 \end{pmatrix}$$

We have now three different bases of $\mathbb{R}^3$:

1. the canonical basis $B = ((1,0,0),(0,1,0),(0,0,1))$,
2. the basis $V = (V_1, V_2, V_3)$,
3. and the basis $E = (\epsilon_1, \epsilon_2, \epsilon_3)$,

So for any vector $x = (x_1, x_2, x_3) \in \mathbb{R}^3$, there are three column matrices associated with x: X, X_V and X_E. We note $P_{BE} = A$ the changing basis matrix between the two bases B and E, we get the relations :

$$X = P_{BE} X_E = A^{-1} X_E$$

$$AX = X_E$$

These relations show that the ante-dual basis E is the right basis to formulate the constraints on the vectors $x \in P$. The polyhedron P is defined by :

$$P = \{(x_1, x_2, x_3) \in (\mathbb{R}^+)^3 \backslash x_1 V_1 + x_2 V_2 + x_3 V_3 \leq b = \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix}\}$$

And

$$x_1 V_1 + x_2 V_2 + x_3 V_3 \leq b \Leftrightarrow AX \leq b \Leftrightarrow X_E \leq b$$

Now we try to maximize the value $\phi(x) = 2L_1(x) - L_2(x)$ over P. If we write this problem with matrices, we get:

$$\begin{cases} \max(2L_1 - L_2)X \\ X = A^{-1} X_E \\ X_E = \begin{pmatrix} X_{E1} \\ X_{E2} \\ X_{E3} \end{pmatrix} \leq \begin{pmatrix} 2 \\ 1 \\ 1 \end{pmatrix} \\ X \geq 0 \end{cases}$$

where L_i is the line matrix identified as the linear form L_i. Then we get $(2L_1 - L_2)X = (2L_1 - L_2)A^{-1}X_E = 2X_{E1} - X_{E2}$. This linear combination is maximized with $X_{E1}^* = 2$ and $X_{E2}^* = 0 = X_{E3}^*$ and on this example, we get that the corresponding $x^* = (0,0,2)$ is positive.
In the same time this choice of X_E induces a natural minimal value reached for the dual problem : From the inequality

$$\max_{x \in P} \phi(x) \leq \max_{\substack{x \in P \\ y \notin P'}} y_1 L_1(x) + y_2 L_2(x) + y_3 L_3(x) \leq y_1 b_1 + y_2 b_2 + y_3 b_3 : y \notin P'$$

the value $x = x^*$ gives:

$$\phi(x^*) \leq y_1 L_1(x^*) + y_2 L_2(x^*) + y_3 L_3(x^*) : y \notin P'$$

and

$$\min_{y \notin P'} y_1 L_1(x^*) = 4 \quad \text{for } y_1 = 2$$

This example uses only basic algebraic calculus. Usually, the strong duality theorem is demonstrated with the Lagrangian relaxation that introduces the notion of local cost for a constraint. The lagrangian technic will be widely studied in the second part of this Thesis and it is interesting to handle only algebraical transformations to get the optimal value here.
It is interesting to mention that a geometric approach of Z_{LP} is to consider each constraint $L_i(x) \leq b_i$ as the half space of $\mathbb{R}^n$ delimited by the hyperplane of equation $L_i(x) = b_i$. P is the intersection of p half spaces with the positive orthant, and introducing the vector $C = (c_1, \ldots, c_n)$, the problem Z_{LP} leads to consider the orthogonal projection of P on $Vect(C)$. Indeed, this approach is not a second dual approach, it is strictly equivalent to those mentioned in 1.4.1 according to the Fréchet-Riesz theorem of representation [52].
To conclude this subsection, I would like to outline that in many computations, duality is merely a way to handle new variables in a scalar product. This way of computing is merely efficient but does not permit to fashion a correct mental representation of the reason why there is no duality gap for a linear problem. Using a scalar product is always a kind of projection on a linear subspace and does not render the complexity of the search of a complete basis adapted to the problem as I try to formulate in the previous example.

1.4.3 System Totally Dual Integral

This subsection is a short description of TDI systems. TDI properties will be exploited in the next chapter in the case of the Maximum Weight Covering Forest 2.

Definition 18 *A system of linear inequalities $Ax \leq b$ is called totally dual integral (TDI) if, for all integer c such that $Z_{LP} = max\{cx : Ax \leq b\}$ is finite, the dual $Z_{LD} = min\{yb : A^t y \geq c, y \in (\mathbb{R}^+)^p\}$ has an integer optimal solution.*

We get the property:

Proposition 19 *If $Ax \leq b$ is TDI and b is integral, then $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ is integral. Hence, for all c for which Z_{LP} is finite, Z_{LP} is integral.*

Operations that preserve TDI systems are described in [12]. An algebraic approach of TDI systems needs to introduce Hilbert basis of a positive cone. Consider a finite set of vectors S of $\mathbb{R}^n$, the positive $cone(S)$ is the set of linear combinations

of elements of S with nonnegative coefficients. The lattice $lat(S)$ is the linear combinations of elements of S with integer coefficients. An Hilbert basis is a minimal finite set of vectors (with integer components) such that every elements of $lat(S) \cap cone(S)$ is a finite combination of elements of H with integer coefficients. Giles and Pulleyblank in [25] have established connections between TDI systems and Hilbert bases, they used the fact that every rational convex cone - a set of type $x : Ax \geq 0$, where A is rational - is generated by a finite Hilbert basis, to show that every integer polyhedron can be described by a TDI system. In our work, the starting point is that the system of inequalities involved in the deterministic covering forest of a graph, is TDI. This central point will be developed in next section.

We gathered in this chapter all the basic tools used to characterize how integer formulations can be relaxed without introducing a duality gap : rank function in matroid, dual formulation and TDIness.

Chapter 2

Maximum Weight Covering Forest

2.1 Introduction

This chapter presents the Maximum Weight Covering Forest. We begin with presenting the deterministic formulation in 2.2.1. The greedy algorithm is recalled in 2.2.2 and we present first results concerning the way an edge is chosen or not during the greedy process in theorem 23. The Two Stage Stochastic formulation of this problem is given in 2.3 and we finish this presentation in 2.4 by a state of the art.

2.2 The deterministic formulation

We consider a graph $G = (V, E)$ where E is a set of edges of cardinality $|E|$, and a cost function c defined on edges. The edges are indexed by $i \in [1, |E|]$ and for any subset F of edges, we call $c(F)$ the sum $\sum_{j \in F} c_j x_j$ where $x_j = 1$ if $j \in F$ and $x_j = 0$ otherwise.

A subset F is said independent if there is no cycle in F. The maximization problem of $c(F)$ for independent F is well known to be connected to matroids and is efficiently solved in the case of a fixed cost value for every edge [48]. When every edge has a fixed value, we say that the problem is **deterministic** and since independent sets are a matroid, the greedy algorithm is efficient to provide the maximization problem of $c(F)$ polynomially. The matroid structure of independent sets involves the fact that the rank function $r(F)$ (the maximum cardinality of any independent subset included in F) is submodular. In this case, the polyhedron associated to dependence constraints is Totally Dual Integral. This is a very

nice property which allows to apply algebraic methods to solve the maximization problem.

2.2.1 Polytopes and problems associated with the deterministic problem

In the deterministic case, we note $n = |E|$ the number of edges in the graph. Considering the graphic matroid of independent subgraphs described in 1.2.2, for any subset S of edges, we introduce the rank function $r(S)$ as the maximal cardinality of the independent subsets of edges in S. Finding a maximum weight forest is formulated by:

$$\mathrm{Z}_{IP} = \begin{cases} max \sum_{j=1}^{n} c_j x_j \\ x \in \{0,1\}^n \\ \sum_{j \in S} x_j \leq r(S) \quad \forall S \subseteq E \end{cases} \tag{2.1}$$

The polytope associated with this problem is:

$$\pi(r) = \{x \in \{0,1\}^n : \sum_{j \in S} x_j \leq r(S) \quad \forall S \subseteq E\} \tag{2.2}$$

The polyhedron associated to the matroid :

$$P(r) = \{x \in (\mathbb{R}^+)^n : \sum_{j \in S} x_j \leq r(S) \quad \forall S \subseteq E\} \tag{2.3}$$

and the linear program :

$$\mathrm{Z}_{LP} = \begin{cases} max \sum_{j=1}^{n} c_j x_j \\ x \in (\mathbb{R}^+)^n \\ \sum_{j \in S} x_j \leq r(S) \quad \forall S \subseteq E \end{cases} \tag{2.4}$$

The dual problem associated with z_{IP} is :

$$\mathrm{Z}_{LD} = \begin{cases} min \sum_{S \subseteq E} r(S) y_S \\ \sum_{S: j \in S} y_S \geq c_j \quad \forall j \in E \\ y_S \geq 0 \quad \forall S \subset E \end{cases} \quad (2.5)$$

We have $Z_{IP} \leq Z_{LP} \leq Z_{LD}$ but Edmonds showed in [48] that $z_{IP} = z_{LD}$ by exhibiting an optimal solution for z_{LD} which is indeed the greedy solution for z_{IP}. The conclusion is that $\pi(r)$ is TDI and the duality gap is equal to 0.

2.2.2 The greedy algorithm

This algorithm is due to Edmonds ([48]).

We call $\mathscr{F}$ the set of independent subsets in E. Rank the elements of E so that $c_1 \geq c_2 \geq \ldots \geq c_n$.

1. Let begin by $J_0 = \emptyset$, $t = 1$.

2. Iteration t : If $c_t \leq 0$ then stop and $S^G = J_{t-1}$.

3. If $c_t > 0$ and $J_{t-1} \cup \{t\} \in \mathscr{F}$, then set $J_t = J_{t-1} \cup \{t\}$.
   If $c_t > 0$ and $S_{t-1} \cup \{t\} \notin \mathscr{F}$ then set $J_t = J_{t-1}$.

4. If $t = N$ stop and $S^G = J_t$

5. Set t to $t + 1$

The greedy solution is $\{x_i : i \in S^G\}$.

We call p the number of chosen edges and $S^G = J_p$.

We now present the main result connecting the greedy solution to the dual problem, and we emphasize the growing closure mechanism. We say that any edge

chosen during the greedy algorithm is *greedy* while any unchosen edge is *covered* . In definition 16, we introduced the central notion of closure or span of a subset : the closure or span of a set A is $sp(A) = \{i \in E : r(A \cup \{i\}) = r(A)\}$.

We briefly describe the greedy algorithm in terms of closure sets. We refer to an edge $x_i \in E$ either directly as x_i or only via its index i. We assume that indexation of edges is equal to their rank. For the edges of the greedy solution for G, let $\{i_1, \ldots, i_t\}$ be the set of edges indexes in decreasing order of edge weights. Denote $J_\tau = \{i_1, \ldots, i_\tau\}, 1 \leq \tau \leq t$ and $\kappa_\tau = sp(J_\tau)$. The first step of the greedy process is the choice of the heaviest weight and we set $J_1 = \{x_1\}$. We call $i_1 = 1$ the first chosen edge and we notice that $\kappa_1 = sp(J_1)$. We now choose the second edge i_2 with the highest rank among all remaining edges such that $i_2 \notin \kappa_1$. Up to this point of construction, obviously $i_2 = 2$ since there is no cycle with only two edges. We build now $J_{\tau+1}$:

The iterative mechanism consists in the choice of $i_{\tau+1}$ with the highest rank among all remaining edges such as $i_{\tau+1} \notin \kappa_\tau$ and $c_{i_{\tau+1}} \geq 0$. It is important to notice that $sp(J_\tau) = sp(\kappa_\tau)$.

Theorem 20 *The dual problem z_{LD} and the greedy solution of z_{IP} have the same objective value. $P(r)$ is an integral polytope, and the system defining $P(r)$ is TDI.*

Proof. We assume that indexation of edges is equal to their rank according to decreasing weights. We dynamically construct both greedy solution and dual solution equals at every step.

According to notations used in 2.2.2, the optimal solution to z_{LD} is :

$y_{\kappa_t} = c_{j_t} - c_{j_{t+1}} \quad for\ t = 1, \ldots, p-1$
$y_{\kappa_p} = c_{j_p}$
$y_S = 0 \quad otherwise$

First, we need to check that for every $j \in N$: $\sum\limits_{S/j \in S} y_S \geq c_j$.
it is obviously the case for every $j_i \in S^G$ since $j_i \in J_t$ for $i \leq t$.
For any j which has not been chosen during the greedy algorithm, j belongs to some $sp(J_t) \setminus sp(J_{t-1})$.
So $\sum\limits_{S/j \in S} y_S = c_{j_t} \geq c_j$.
Secondly, we need to compute the sum : $\sum\limits_{S \subseteq E} r(S) y_S = \sum\limits_{t=1}^{p-1} t(c_{j_t} - c_{j_{t+1}}) + p c_{j_p}$
this leads, when splitting in two sums and re-indexing to $\sum\limits_{S \subseteq E} r(S) y_S = \sum\limits_{t=1}^{p} c_{j_t}$.
This shows that z_{IP} and z_{LD} have the same objective value. More precisely, the greedy solution to z_{IP} and z_{LD} have the same objective value. We conclude that

the set of inequalities defining $P(r)$ is TDI, and $P(r)$ is an integral polytope. ■

It is important to remark the following points:

- The dual variable chosen at step t is the one associated with the *heaviest* constraint $\sum_{j \in \kappa_t} x_j \leq r(\kappa_t)$ in the sense that among all subsets S of rank t, the linear forms introduced in 1.4.1 verify $L_{\kappa_t}(x_1, \ldots, x_n) \geq L_S(x_1, \ldots, x_n)$
- The dual variable depends only on the current chosen weighted edge and the next one. That means indeed that the greedy algorithm will not change if we modify any weighted edge with a smaller cost than those examined at step t, independently of any geometrical consideration in the graph.

This second remark leads to introduce the next definition and the following theorem:

Definition 21 *For a given cost $c \in \mathbb{R}^N$ on G and $b \in \mathbb{R}$, we note $U(b) = \{x_i : c_i \geq b\}$*

Lemma 22 *For any graph G of cardinal N, consider specifically any edge x with cost $c \in \mathbb{R}$ considered as a variable, and assume that every other edges $(x_1, \ldots, x_{N-1})$ have a given cost $(c_1, \ldots, c_{N-1})$. There exists a threshold $b \in \mathbb{R}^+$ such that if $c > b$, then x belongs to the greedy solution applied to G, while x is covered if $c < b$. The threshold b is a function of $(c_1, \ldots, c_{N-1})$. When $c = b$, the status of x can be greedy or covered according to the choice of x or another edge of same weight during the greedy algorithm.*

Proof. We still assume that the indexation of edges $x_1, \ldots, x_{N-1}$ is equal to their rank according to a decreasing weight. We dynamically construct a greedy solution on $\tilde{G} = G \setminus \{x\}$ without taking into account the specific edge x. We get a finite sequence of sets $\tilde{J}_1, \ldots, \tilde{J}_p$ and of corresponding closures $\tilde{\kappa}_1, \ldots, \tilde{\kappa}_p$ of $\tilde{G}$. Remark that $\tilde{\kappa}_1, \ldots, \tilde{\kappa}_p$ is a strictly increasing sequence for inclusion of subsets such that $G \setminus \{x\} = \tilde{\kappa}_p$.
We reintroduce x as a new edge in the graph $\tilde{G}$. In the case where $x \notin \tilde{\kappa}_p$, that means that as soon as $c > 0$ then x would be chosen during the greedy algorithm applied properly on the whole graph G, so that $b = 0$.
In the case where $x \in \tilde{\kappa}_p$ that means that there exists a particular step τ such that $x \in sp(\tilde{J}_{\tau+1})$ and $x \notin sp(\tilde{J}_\tau)$. The threshold b is equal to $c_{i_{\tau+1}}$. In the case where $c = c_{i_{\tau+1}}$ occurs, then x and $x_{i_{\tau+1}}$ have the same weight (and perhaps several other edges) and x or $x_{i_{\tau+1}}$ can be indifferently chosen during the greedy algorithm but not both together. ■

Theorem 23 *$b(c_1, \ldots, c_{N-1})$ is a continuous function of $(c_1, \ldots, c_{N-1})$*

Proof. We consider the same mechanism as in lemma 22 to get a first threshold $b = b(c_1, \ldots, c_{N-1})$ when removing x from G. We begin to notice that during the greedy algorithm, at every step, when choosing the τth edge of the greedy solution, $J_\tau \subseteq U(c_{i_\tau}) \subseteq \kappa_\tau$.
We fix $\epsilon > 0$ and for $c = (c_1, \ldots, c_{N-1}) \in \mathbb{R}^{N-1}$, we consider a small perturbation $c' = (c'_1, \ldots, c'_{N-1})$ such that $|c_i - c'_i| < \epsilon \quad \forall i \in [1, \ldots, N-1]$. We note $U'(b) = \{x_i / c'_i \geq b\}$, the set of edges whose slightly modified weights are greater than b.
Obviously, if x is not covered by any independent subset in $U'(b)$, that means that b will decrease. Conversely, if $U'(b)$ contains some new independent subsets that covers x, it will possibly enforce b to increase. We begin to explain how the variation of the threshold is lower bounded: $U(b)$ contains a subset J_τ such that $x \in sp(J_\tau)$ and $U(b) \subset U'(b - 2\epsilon)$. Then $U'(b - 2\epsilon)$ contains a subset $J'_{\tau'}$, step of the greedy algorithm applied to $\tilde{G}$ with c' cost and such that $x \in sp(J'_{\tau'})$. This proves that $b(c') \geq b - 2\epsilon$.
For the upper bound of the variation, we see that since $U(b + \epsilon)$ does not contain any independent set covering x with cost c, and $U'(b+2\epsilon) \subseteq U(b+\epsilon)$ then $U'(b+2\epsilon)$ does not contain any independent set covering x in G with c' cost. This proves that $b(c') \leq b(c) + 2\epsilon$. We conclude that $|b(c') - b(c)| \leq 2\epsilon$, and b is a continuous function of c. ■

2.3 The Two Stage Stochastic formulation

The problem that we introduce now is to understand what happens when cost function is not deterministic but follows a discrete probability distribution π. In our problem, the edges are split into two subsets $E = X \cup Y$. In stochastic programming, the first subset X has a deterministic cost function, this set is called first stage, we set $card(X) = n$, whereas in second subset called second stage, $card(Y) = q$. We notice $N = n + q$ the cardinal of the whole set of edges in G.
The cost function depends on $K \geq 2$ scenarios and the cost values are given by a probability distribution $\pi = (\pi_1, \ldots, \pi_K)$. The formulation of this problem is:

$$z_{IP} = \begin{cases} max \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \pi_k \sum_{j \in Y} c_j(k) z_j^k \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_j^k \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E \\ (x, z^k) \in \{0,1\}^n \times \{0,1\}^q, k \in \{1, \ldots, K\}. \end{cases} \tag{2.6}$$

We outline that any first stage edge is associated with a single binary variable x_j, while a second stage edge is associated with K binary variables z_j^k. In the deterministic case, we had only $n = N = |E|$. We introduce the notations:

The linear program associated to (6.2) is:

$$z_{LP} = \begin{cases} max \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \pi_k \sum_{j \in Y} c_{jk} z_{jk} : \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E \\ (x, z_k) \in [0,1]^n \times [0,1]^q, k \in \{1, \ldots, K\}. \end{cases} \tag{2.7}$$

and its LP-dual is:

$$z_{LD} = \begin{cases} min \sum_{k=1}^{K} \sum_{S \subseteq E} r(S) y_{S,k} : \\ \sum_{k=1}^{K} \sum_{S \subseteq E : i \in X \cap S} y_{S,k} \geq c_i, \quad i \in X \\ \sum_{S \subseteq E : j \in Y \cap S} y_{S,k} \geq \pi_k c_{jk}, \quad j \in Y, k \in \{1, \ldots, K\} \\ y_{S,k} \geq 0, \quad k \in \{1, \ldots, K\}, \quad S \subseteq E. \end{cases}$$

(2.8)

Remark that in some parts of this work, we omit the stochastic vector π by considering directly a new cost c_{jk} which is indeed the product of π_k by c_{jk}.

2.4 A general introduction of two stages problems

The problem 6.2 belongs to the general model class :

$$z_P = \begin{cases} \max_{x \in X} c^T x + \mathbb{E}[Q(x, \xi)] \\ s.t. Ax \leq b, x \geq 0 \end{cases} \tag{2.9}$$

where $Q[x, \xi]$ is the optimal value of the second-stage problem, and $\mathbb{E}$ stands for expectation :

$$\text{Q[x,}\xi] = \begin{cases} \max_{z \in Y} q^T z \\ s.t. Tx + Wz \leq h, z \geq 0 \end{cases}$$

(2.10)

where $\xi = (q, h, T, W)$ are the random data of the second-stage. X and Y are an integer domains, which is indeed the worst case to handle. The relaxed problem 2.7 belongs to the the same class of problems, but, due to the fact that X and Y are continuous and that the number of scenarios is finite, several results can be obtained easily. An introduction concerning these problems is described in [55]. A survey for specific strategies in case of Two-Stage stochastic Integer programming is given in [53]. A tutorial on Stochastic Programming and Applications is given in [54]. We begin by explaining how this relaxed problem can be treated and why the integer version is very difficult to solve since we loose convexity properties.
Considering any fixed value of x in 2.7, the second-stage problem is a linear problem whose dual is given by:

$$\begin{cases} \min\limits_{\pi} \pi^T(h - Tx) \\ s.t. W^T\pi \geq q \end{cases} \quad (2.11)$$

Observe that this formulation is compatible with strong duality results recalled in 1.4.1. Unless both problems are infeasible, they have the same optimal value. If we introduce the function $S_q(\chi) = sup\{q^T z : Wz \leq \chi, z \geq 0\}$ and $\Pi(q) = \{\pi : W^T\pi \geq q\}$, then

$$S_q(\chi) = \inf_{\pi \in \Pi(q)} \pi^T \chi$$

and

$$S_q(h - Tx) = Q(x, \xi).$$

The result is that for any realization of ξ, the function $Q(., \xi)$ is convex because S_q is a polyhedral function. [1] Moreover, since the probability distribution has a finite support, the problem 2.7 can be considered as a large-scale deterministic linear problem.
Unfortunately, as soon as integer variables are part of the formulation in second stage, continuity and consequently convexity properties can disappear. Consider the following problem:

$$\begin{cases} \max 30x_1 + 30x_2 + 30x_3 + 10y_1 + 10y_2 + 10y_3 \\ x_1 + x_2 + y_1 \leq 2 \\ x_2 + x_2 + y_2 \leq 2 \\ x_3 + x_1 + y_3 \leq 2 \\ x_i \geq 0, i \in \{1, 3\} \\ y_i \in \{0, 1\}, i \in \{1, 3\} \end{cases} \quad (2.12)$$

It is easy to compute that for $x_1 = \frac{1}{2}, x_2 = \frac{1}{2}, x_3 = \frac{1}{2} + \epsilon$ where $\epsilon > 0$, the optimal solution is $y_1 = 1, y_2 = 0, y_3 = 0$ with the optimal value $55 + 30\epsilon$, while for

[1] A polyhedral function f is proper convex and lower semi-continuous, its domain is a closed convex polyhedron and f is piecewise linear

$x_1 = \frac{1}{2}, x_2 = \frac{1}{2}, x_3 = \frac{1}{2} - \epsilon$, the optimal solution is $y_1 = 1, y_2 = 1, y_3 = 1$ with the optimal value $75 - 10\epsilon$.

Several technics have been used to deal with integer stochastic programs. We mainly distinguish exact and approximation approaches. In exact approaches, branch and bound methods are presented in [3]. In this type of approach, the main difficulty is to avoid exponential enumeration of the solution space. Another type of exact approach is presented in [37], where superadditive duality properties are exploited. Unlike in branch and bound algorithms, the column elimination (setting some variables x_j to 0), is based on a level-set approach. This approach can be understood as a increasing improvement of the global objective function, by adding some sub-objective value -called the value function- to the expected value of a second stage function, whose variables are called "tender variables".

Among approximation methods, it is possible to build a continuous convex hull of the second stage value function. This approach is used in case of a simple recourse situation [2] and analyzed in [28]. The branch and bound technic is often used in approximation too, this type of approach uses generally optimal cuts [41] and can be coupled with approximation of the second stage value function [8].

Another way to classify the approximation methods is to consider whether the exploration scheme mixes first stage variables with two stage variables, or only fixes first stage variables and explores some relative two stage variable issues. Benders' like method are adapted to two stage stochastic formulations because of the natural shape of the constraint matrix. In Benders' method, selecting nice and complicating variables is a delicate balance between complexity of the primal problem and the number of constraints to relax. In Mixed Integer Programs (MIP), one can choose to keep all integer variables into the main program or, conversely, to put them into the lagrangian relaxation formulation. In the case of pure integer variables, the dual lagrangian problem associated to the relaxed variables occurs after the relaxation of complicating variables [9] [29].
When mixing first and second stage variables, the starting point is to introduce "non anticipativity constraints". These constraints state the property that any first stage variable has the same value in every scenario [55]. Then, it is possible to use Lagrangian multipliers. We will explain why these constraints are inappropriate in our approach 3.2.

Some unclassical approaches adopt a different point of view [53] by decomposing the problem after translating it into another space variable (graver test set)

Some specific situations have been studied more specifically, trying to extend properties of the deterministic formulations to the case of stochastic versions. The present work is connected with this class of approach. The case of totally unimodular programs is characteristic of the efforts of research in this domain. A matrix

[2] a simple recourse means that each second stage variable is bounded separately by two values; it follows that the recourse matrix W is $[I, -I]$

A is said to be totally unimodular if and only if for any squared submatrix $A' \subset A$, $det(A') \in \{0, 1, -1\}$. That property ensures that for any $b \in \mathbb{Z}^m$, the polyhedron $P = \{x \in (\mathbb{R}^+)^n \, Ax \leq b\}$ is integer [48], [4], [13]. Totally unimodular matrices are present in network flows and are studied in [16]. Interval matrices are Totally Unimodular [22]. In [36], Kong, Shabbir and Schaeffer study an extension of totally unimodular matrices in the case of a constraint matrix B associated to the program 2.4.

$$B = \begin{pmatrix} T^1 & W^1 & & & \\ T^2 & & W^1 & & \\ \vdots & & & \ddots & \\ T^K & & & & W^K \end{pmatrix}$$

They extend the definition of totally unimodular matrices by coming back to the fundamental root of the nature of a unimodular matrix exposed in the Ghouila-Houri theorem [24]:

Theorem 24 *A $\{0, 1, -1\}$ $m \times n$ matrix A is TU if and only if for every $J \subset \{1, \ldots, n\}$, there exists a partition (J^1, J^2) of J such that*

$$\left| \sum_{j \in J^1} A_{ij} - \sum_{j \in J^2} A_{ij} \right| \leq 1, for \quad i = 1, \ldots, m..$$

The authors introduce some synchronization between technology submatrices T^i according to a specific vector v, and then define conditions over recourse matrices W^j in order to satisfy similar conditions that those given in Ghouila-Houri theorem. The aim we tried to reach here is similar to this approach. In the deterministic case, the maximal covering forest is solved in polynomial time since the formulation gives a TDI system. We try to analyze in which stochastic cases, TDI properties are preserved and how to proceed when it is not the case.

Chapter 3

Two stage problems with only two scenarios

3.1 Introduction

In this chapter, we study the case of a two-stage problem with only two scenarios. This situation leads to introduce the formal split of the cost for the first level edges. In 3.2, any first level cost is split in many subcosts corresponding to each scenario. These subcosts give a corresponding status for any first level edge considered in a specific scenario. The central point is that we do not create many decision variables associated to each first stage level, which is far different from introducing non anticipativity constraints. In 3.3, we prove that a two-stage problem with only two scenarios is always TDI in three steps: first with only one first stage edge, secondly with only two first stage edges, and finally by induction on the number of first stage edges.

3.2 Formal split of the cost of a first stage edge

Since the cost of second stage edges change with scenarios, while first stage edges costs remain the same, when one specific scenario occurs, we apply a greedy algorithm not with the whole cost of first stage edge, but only with a fractional part as described below:
For $E = X \cup Y$ first and second stage edges, and for (c, c_k) weights vectors with $c \in \mathbb{R}^n$, $c_k \in \mathbb{R}^q, k = 1, \ldots, K$.
Consider a split of the form

$$c_i = \sum_{k=1}^{K} c_i^k, \ i \in X$$

with

$$c_i^k \geq 0,\ i \in X$$

or an equivalent vector formulation:

$$c = \sum_{k=1}^{K} c^k$$

with

$$c^k \geq 0$$

For $k \in \{1, \ldots, K\}$ we consider the sets $\{i_1^k, \ldots, i_{t_k}^k\}$ of indexes in the order they are picked by the greedy algorithm applied for each (c^k, c_k) cost vector.

By κ_τ^k we denote the spans of the following subsets of the edge sets in the greedy sequence

$$\{i_1^k, \ldots, i_{t_k}^k\},\ \tau = 1, \ldots, t_k,\ k = 1, \ldots, K.$$

Remark

$$r(\kappa_\tau^k) = \tau,\ \tau = 1, \ldots, t_k,\ k = 1, \ldots, K.$$

We introduce a specific condition on first stage edges in every scenario:

Definition 25 *For the individual weight vectors (c^k, c_k), $k = 1, \ldots, K$, if each first stage edge is either always or never picked simultaneously in every scenario by the greedy algorithm, we say that the status of first stage edges is uniform.*

In chapter 2 of [55], the mechanism of non anticipativity constraints is developed. Since the first level edges variables are replaced by K equivalent distinct new variables, we need to give some global coherence by adding new constraints as $x_i^1 = \ldots = x_i^K$ for all $i \in \{1, \ldots, n\}$. These constraints are called non-anticipativity constraints. Unfortunately, these constraints are inefficient to turn the system into a TDI one. They only set the same fractional value to all first stage edges. For instance, the simple example given in annex A with Porta and nonanticipativity constraints still gives several fractional solutions.

Theorem 26 *Assume that for any c, there exists a split $c = \sum_{k=1}^{K} c^k$ with $c^k \geq 0$ fulfilling the condition of definition* (25)*; then the system*

$$\left\{ \begin{array}{l} (x, z_1, \ldots, z_K) \in [0,1]^n \times [0,1]^q \times \ldots \times [0,1]^q : \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E \end{array} \right. \quad (3.1)$$

is totally dual integral.

Proof. Let $(c, c_1, \ldots, c_K)$ be an arbitrary weight vector with $c \in \mathbb{R}^n$, $c_k \in \mathbb{R}^q$, $k = 1, \ldots, K$. Consider the two stage stochastic maximum problem:

$$\begin{cases} max \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \sum_{j \in Y} c_{jk} z_{jk} \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E \\ (x, z_k) \in \{0,1\}^n \times \{0,1\}^q, k \in \{1, \ldots, K\}. \end{cases} \tag{3.2}$$

Consider a split for c fulfilling condition (25) and for $k = 1, \ldots, K$, put $x_i^k = 1, z_{ik} = 1$ if the greedy algorithm picks $i \in X$ respectively $i \in Y$, under the vector (c^k, c_k), and $x_i^k = 0, z_{ik} = 0$ otherwise. We obtain :

$$\sum_{k=1}^{k=K} \sum_{j \in X} c_j^k x_j^k + \sum_{k=1}^{k=K} \sum_{j \in Y} c_{jk} z_{jk} = \sum_{j \in X} (\sum_{k=1}^{k=K} c_j^k) x_j + \sum_{k=1}^{k=K} \sum_{j \in Y} c_{jk} z_{jk} = \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \sum_{j \in Y} c_{jk} z_{jk} \tag{3.3}$$

The feasibility of the scenario specific solutions

$$\sum_{j \in S \cap X} x_j^k + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E$$

implies the feasibility of the two-stage model

$$\sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E$$

Now turn to the dual of the LP relaxation aiming at the construction of an optimal solution whose objective value coincides with (3.3):

$$\begin{cases} min \sum_{k=1}^{K} \sum_{S \subseteq E} r(S) y_{S,k} : \\ \sum_{k=1}^{K} \sum_{S \subseteq E : i \in X \cap S} y_{S,k} \geq c_i, \quad i \in X \\ \sum_{S \subseteq E : j \in Y \cap S} y_{S,k} \geq c_{jk}, \quad j \in Y, k \in \{1, \ldots, K\} \\ y_{S,k} \geq 0, \quad k \in \{1, \ldots, K\}, \quad S \subseteq E. \end{cases}$$

(3.4)

For each $k = 1, \ldots, K$ let, according to the choice $\{i_1^k, \ldots, i_{t_k}^k\}$, $\tau = 1, \ldots, t_k$, $k = 1, \ldots, K$ and in descending order, $\hat{c}_{i_1^k} \geq \ldots \geq \hat{c}_{i_{t_k}^k} \geq 0$ be the weights of the edge picked by the greedy algorithm runned on the instance with edge weights

(c^k, c_k). According to the deterministic proof of the greedy solution, for the edge sets $S = \kappa^k_\tau,\ \tau = 1, \dots, t_k,\ k = 1, \dots, K$ we put

$$y_{S,k} = \hat{c}_{i^k_\tau} - \hat{c}_{i^k_{\tau+1}},\ \tau = 1, \dots, t_k - 1 \text{ and } y_{S,k} = \hat{c}_{i^k_{t_k}},\ \tau = t_k.$$

For the remaining $S \subseteq E$ and $k = 1, \dots, K$, we put $y_{S,k} = 0$. To check feasibility, fix some $i \in X$, then for each $k = 1, \dots, K$, there exists a unique index τ^* with $i \in \kappa^k_{\tau^*+1} \setminus \kappa^k_{\tau^*}$. It holds:

$$\sum_{S \subseteq E: i \in X \cap S} y_{S,k} = \sum_{\tau^*}^{t_k} y_{\kappa^k_\tau, k} = \sum_{\tau^*}^{t_k} (\hat{c}_{i^k_\tau} - \hat{c}_{i^k_{\tau+1}}) = \hat{c}_{i^k_{\tau^*}} \geq c^k_i.$$

Summing up over k yelds

$$\sum_{k=1}^{K} \sum_{S \subseteq E: i \in X \cap S} y_{S,k} \geq \sum_{k=1}^{K} c^k_i = c_i.$$

For $i \in Y$ and $k \in \{1, \dots, K\}$ again there exists a unique index τ^* with $i \in \kappa^k_{\tau^*+1} \setminus \kappa^k_{\tau^*}$, and we have:

$$\sum_{S \subseteq E: i \in Y \cap S} y_{S,k} = \sum_{\tau^*}^{t_k} y_{\kappa^k_\tau, k} = \sum_{\tau^*}^{t_k} (\hat{c}_{i^k_\tau} - \hat{c}_{i^k_{\tau+1}}) = \hat{c}_{i^k_{\tau^*}} \geq c_{ik}.$$

Non-negativity of the dual solution is immediate. If $i \notin \kappa^k_{t_k}$, then its edge weight is non-positive, and the dual constraint involving i is satisfied. For the dual objective, it holds

$$\begin{aligned}
&\sum_{k=1}^{K} \sum_{S \subseteq E} r(S) y_{S,k} && (3.5)\\
&= \sum_{k=1}^{K} \sum_{\tau=1}^{t_k} r(\kappa^k_\tau) y_{\kappa^k_\tau, k} = \sum_{k=1}^{K} \sum_{\tau=1}^{t_k - 1} \tau(\hat{c}_{i^k_\tau} - \hat{c}_{i^k_{\tau+1}}) + t_k \hat{c}_{i^k_{t_k}} = \sum_{k=1}^{K} \sum_{\tau=1}^{t_k} \hat{c}_{i^k_\tau} && (3.6)\\
&= \sum_{k=1}^{K} (\sum_{i \in X} c^k_i x^k_i + \sum_{i \in Y} c_{ik} z_{ik}) = \sum_{k=1}^{K} (\sum_{i \in X} c^k_i x_i + \sum_{i \in Y} c_{ik} z_{ik}) && (3.7)\\
&= \sum_{i \in X} (\sum_{k=1}^{K} c^k_i) x_i + \sum_{k=1}^{K} \sum_{i \in Y} c_{ik} z_{ik} && (3.8)\\
&= \sum_{i \in X} c_i x_i + \sum_{k=1}^{K} \sum_{i \in Y} c_{ik} z_{ik} && (3.9)
\end{aligned}$$

which coincides with (3.3). Hence the system in question is totally dual integral.

■

3.3 Two stage problem with only two scenarios

The case $K = 2$ is very different from the case $K \geq 3$. We prove in this section that in the case $K = 2$, the optimal value of the problem is integer. The main idea is to prove the existence of a correct split of first stage edge costs according to (25) by induction on the number of first stage edges. Yet, the case of only one first stage edge gives a basic settlement for any value of K, and the case of only two first stage edges is a central point in the proof. When there exists a split $c = \sum_{k=1}^{K} c^k$ with $c^k \geq 0$ fulfilling the condition of definition (25), we say that the first stage edge i is *'covered'* if the split turns to $x_i^k = 0$ $k = 1, \ldots, K$ when applying greedy algorithms. Conversely, we say that the first stage i is *'chosen'* if the split turns to $x_i^k = 1$ $k = 1, \ldots, K$.

3.3.1 The case of only one edge in the first level

We consider the case where only one edge x_1 belongs to the first stage.

Theorem 27 *With only one edge in the first stage and $K = 2$, the primal problem z_{IP} (6.2) and dual problem z_{LP} (2.3) have the same integer value. This entails that the system is TDI.*

Proof. It suffices to prove that it is possible to correctly split the cost c_1 into two parts in order to get the same status in both scenarios.

According to lemma 22, there exist two thresholds $b_1^1 = b_1^1(c_{11}, \ldots, c_{q1})$ and $b_1^2 = b_1^2(c_{12}, \ldots, c_{q2})$ that determine the status of x_1 in each scenario.
In the case where $b_1^1 + b_1^2 \leq c_1$ then it is possible to split c_1 with respect to $b_1^1 \leq c_1^1$ and $b_1^2 \leq c_1^2$.
In the case where $b_1^1 + b_1^2 \geq c_1$ then it is possible to split c_1 with respect to $b_1^1 \geq c_1^1$ and $b_1^2 \geq c_1^2$.
From the point of view of a single scenario, these different cases can be summarized into one single criteria: for $c_1^1 \in [min(b_1^1, c_1 - b_1^2), max(b_1^1, c_1 - b_1^2)]$, the status of x_1 into both scenarios during greedy algorithm is the same. ■

The case of only one edge in the first stage with any number of scenarios can easily be answered in the same manner:

Theorem 28 *In the case of any number of scenarios $K \geq 2$, with only one edge in the first stage, the primal problem z_{IP} (6.2) and dual problem z_{LP} (2.3) have the same integer value. This entails that the system is TDI.*

Proof. It suffices to prove that it is possible to correctly split the cost c_1 into K parts in order to get the same status in every scenario.

According to lemma 22, there exist K thresholds $b_1^k = b_1^k(c_{1k}, \ldots, c_{qk})$, $k = 1, \ldots, K$. In the case where $\sum_{k=1}^{K} b_1^k \leq c_1$, it is possible to split c_1 with respect to

$b_1^k \leq c_1^k$, $\forall k$

In the case where $\sum_{k=1}^{K} b_1^k \leq c_1$ then it is possible to split c_1 with respect to $b_1^k \geq c_1^k$, $\forall k$

■

3.3.2 The case of two edges in the first stage

We consider the case where two edges x_1 and x_2 belong to the first stage with respectively costs c_1 and c_2. We proceed in the same way as with one single edge by splitting fixed costs into two parts: $c_1 = c_1^1 + c_1^2$ and $c_2 = c_2^1 + c_2^2$.

Theorem 29 *With only two edges in first stage and $K = 2$, the primal problem z_{IP} (6.2) and dual problem z_{LP} (2.3) have the same integer value. This entails that the system is TDI.*

Proof.

In the proof of theorem 27, we have seen that an accurate split of the cost of one edge is given by a compact interval $[min(b_1^1, c_1 - b_1^2), max(b_1^1, c_1 - b_1^2)]$, where b_1^1 and b_1^2 are two continuous functions of the other costs of all edges -independently of their stage.

Since there are only two scenarios, any split of the cost $c_2 = c_2^1 + c_2^2$ can be interpreted as the variation of a single parameter c_2^1. That means that b_1^1 (respectively b_1^2) can be seen as a function depending of a simple variable c_2^1 and several fixed parameters $c_2, c_{11}, \ldots, c_{q1}$ (respectively $c_2, c_{12}, \ldots, c_{q2}$):

$$b_1^1 = b_1^1(c_2^1, c_{11}, \ldots, c_{q1})$$

and

$$b_1^2 = b_1^2(c_2 - c_2^1, c_{12}, \ldots, c_{q2}).$$

Considering the standalone variation of the value $c_2^1 \in [0, c_2]$, we get two continuous functions $f_1(c_2^1) = min(b_1^1, c_1 - b_1^2)$ and $g_1(c_2^1) = max(b_1^1, c_1 - b_1^2)$, defining the boundary of a never empty area of the two dimensional space for variables $(c_1^1, c_2^1) \in [0, c_1] \times [0, c_2]$ where x_1 has a common status in scenario $\mathscr{S}_1$ and $\mathscr{S}_2$ i.e. $x_1^1 = x_1^2$. See figure 3.3.2

Functions f_1 and g_1 define two continuous parametric curves connecting respectively the points $(f_1(0), 0)$ to $(f_1(c_2), c_2)$ and $(g_1(0), 0)$ to $(g_1(c_2), c_2)$ in the space $(c_1^1, c_2^1) \in [0, c_1] \times [0, c_2]$.

The same analysis with the second first stage edge leads to introduce two similar thresholds:

$$b_2^1 = b_2^1(c_1^1, c_{11}, \ldots, c_{q1})$$

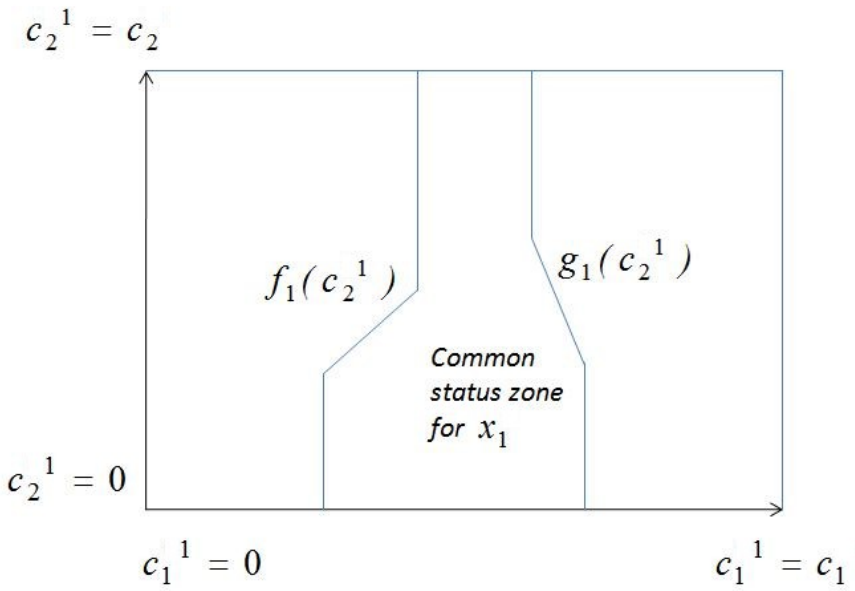

Figure 3.1: common status for x_1 between two continuous parametric curves

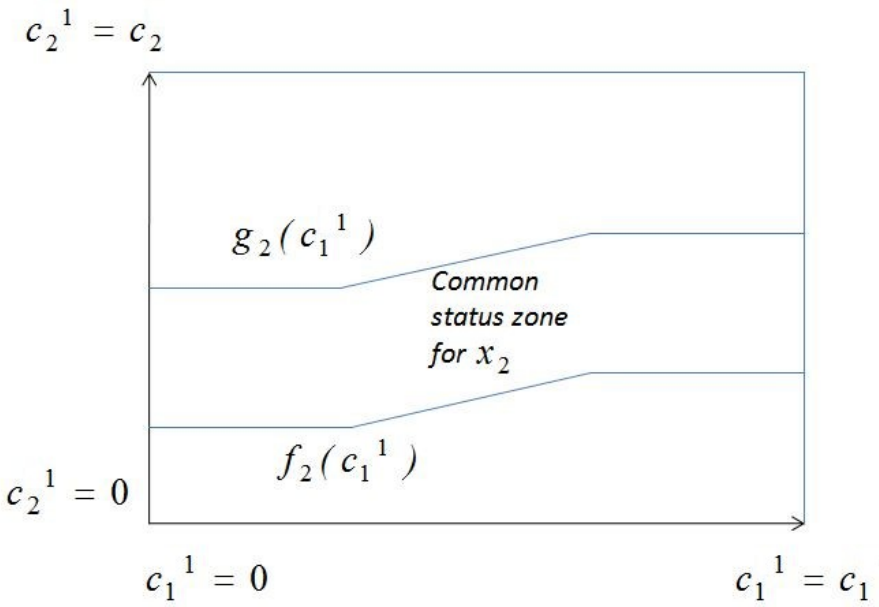

Figure 3.2: common status for x_2 between two continuous parametric curves

and

$$b_2^2 = b_2^2(c_1 - c_1^1, c_{12}, \ldots, c_{q2})$$

A split of c_2 between these two thresholds gives the same status of x_2 in both scenarios.
We then consider in the same manner two continuous functions $f_2(c_1^1) = min(b_2^1, c_2 - b_2^2)$ and $g_2(c_1^1) = max(b_2^1, c_2 - b_2^2)$ defining the boundary of a non-empty area of the same two dimensional space for variables $(c_1^1, c_2^1) \in [0, c_1] \times [0, c_2]$, where x_2 has a common status in scenario $\mathscr{S}_1$ and $\mathscr{S}_2$.
Functions f_2 and g_2 define two continuous parametric curves connecting respectively the points $(0, f_2(0))$ to $(c_1, f_2(c_1))$ and $(0, g_2(0))$ to $(c_1, g_2(c_1), 0)$ in the space $[0, c_1] \times [0, c_2]$. See figure 3.2
According to the theorem of intermediate values for continuous functions, there exist crossing values for curves (f_1, f_2), (g_1, f_1), (f_1, g_2) and (g_1, g_2) which define a zone with continuous parametric curves for boundary and where x_1 and x_2 have simultaneously the same status in $\mathscr{S}_1$ and $\mathscr{S}_2$. See figure 3.3

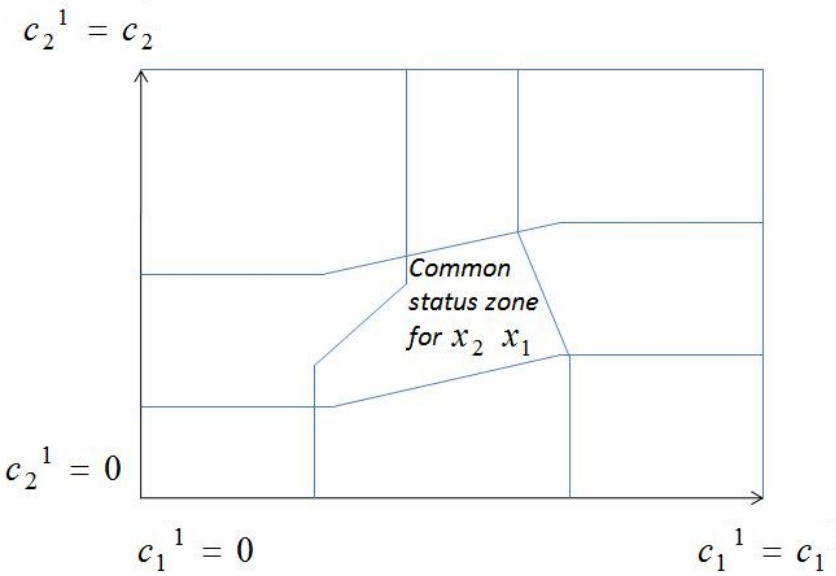

Figure 3.3: common status for x_1 and x_2

■

3.3.3 The case of any number of first stage edges with only two scenarios

Theorem 30 *With only two scenarios, the primal problem z_{IP} and dual problem z_{LP} have the same integer value and the system is TDI.*

Proof. We prove this theorem by an induction based on the number of first stage edges. We claim the following statement:
$H(n) \Leftrightarrow$ "with n edges in the first stage, there exist a correct split of the costs of every edges of first stage in terms of $c_1 = c_1^1 + c_1^2, \ldots, c_n = c_n^1 + c_n^2$ such that these edges get the same respective uniform status in scenario $\mathscr{S}_1$ and $\mathscr{S}_2$ with respect to condition (25). When focusing on the part of these splits concerning the first scenario, the correct split is given by $(c_1^1, \ldots, c_n^1) \in \Omega_{(1,\ldots,n)}$ where $\Omega_{(1,\ldots,n)} \subset [0, c_1] \times \ldots \times [0, c_n]$ is a regular compact zone whose boundary is defined by regular (continuous) parametric hypersurfaces.
These hypersurfaces are specific thresholds of the kind

$$f_i(c_1^1, \ldots, c_{i-1}^1, c_{i+1}^1, \ldots, c_n^1, c_{11}, \ldots, c_{1q}, c_{11}, \ldots, c_{q1}, c_{12}, \ldots, c_{q2}) = min(b_i^1, c_i - b_i^2),\ i \in \{1, \ldots, n\}$$

and

$$g_i(c_1^1, \ldots, c_{i-1}^1, c_{i+1}^1, \ldots, c_n^1, c_{11}, \ldots, c_{q1}, c_{12}, \ldots, c_{q2}) = max(b_i^1, c_i - b_i^2),\ i \in \{1, \ldots, n\}$$

or intersections of such thresholds."
The case of $n = 2$ has been considered in section 3.3.2.
Assume that $H(n)$ is true for some value n, we consider a graph G with $n + 1$ first stage edges and q second stage edges. We split the value c_{n+1} into $c_{n+1} = c_{n+1}^1 + c_{n+1}^2$. To every value c_{n+1}^1, by application of $H(n)$, it corresponds a regular compact zone $\Omega(c_{n+1}^1)$ into which boundaries are given by specific thresholds of the kind :

$$f_i(c_1^1, \ldots, c_{i-1}^1, c_{i+1}^1, \ldots, c_n^1, c_{n+1}^1, c_{11}, \ldots, c_{q1}, c_{12}, \ldots, c_{q2}) = min(b_i^1, c_i - b_i^2)$$

and

$$g_i(c_1^1, \ldots, c_{i-1}^1, c_{i+1}^1, \ldots, c_n^1, c_{n+1}^1, c_{11}, \ldots, c_{q1}, c_{12}, \ldots, c_{q2}) = max(b_i^1, c_i - b_i^2)$$

or intersections of such thresholds.
Since functions involved of type f or g are min or max of continuous functions of the kind of b_i^1 or b_i^2 defined in 23, the collection $\Omega_{(1,\ldots,n)} = \{\Omega(c_{n+1}^1), c_{n+1}^1 \in [0, c_{n+1}]\}$ defines a continuous zone into which all first stage edges $x_1, \ldots, x_n$ have respectively a uniform status in both scenarios according to condition (25).
Consider now the proper thresholds for x_{n+1} given by

$$f_{n+1}(c_1^1, \ldots, c_n^1, c_{11}, \ldots, c_{q1}, c_{12}, \ldots, c_{q2}) = min(b_{n+1}^1, c_{n+1} - b_{n+1}^2)$$

and

$$g_{n+1}(c_1^1, \ldots, c_n^1, c_{11}, \ldots, c_{q1}, c_{21}, \ldots, c_{q2}) = max(b_{n+1}^1, c_{n+1} - b_{n+1}^2)$$

These two functions define respectively two regular (continuous) parametric hypersurfaces. Between these two hypersurfaces, x_{n+1} has a common status in scenario $\mathscr{S}_1$ and $\mathscr{S}_2$. If we call Ω_{n+1} the subset of $[0, c_1] \times \ldots \times [0, c_{n+1}]$ between these two hypersurfaces, then the crossing part $\Omega_{(1,\ldots,n)} \cap \Omega_{n+1} = \Omega_{(1,\ldots,n+1)}$ is a non-empty zone where all $n+1$ first stage edges have a uniform status in both scenarios. The boundary of this intersection is of the same kind as those described in $H(n)$ and that ends the proof for $H(n+1)$. For illustration with $n = 3$, see figure (3.4). ■

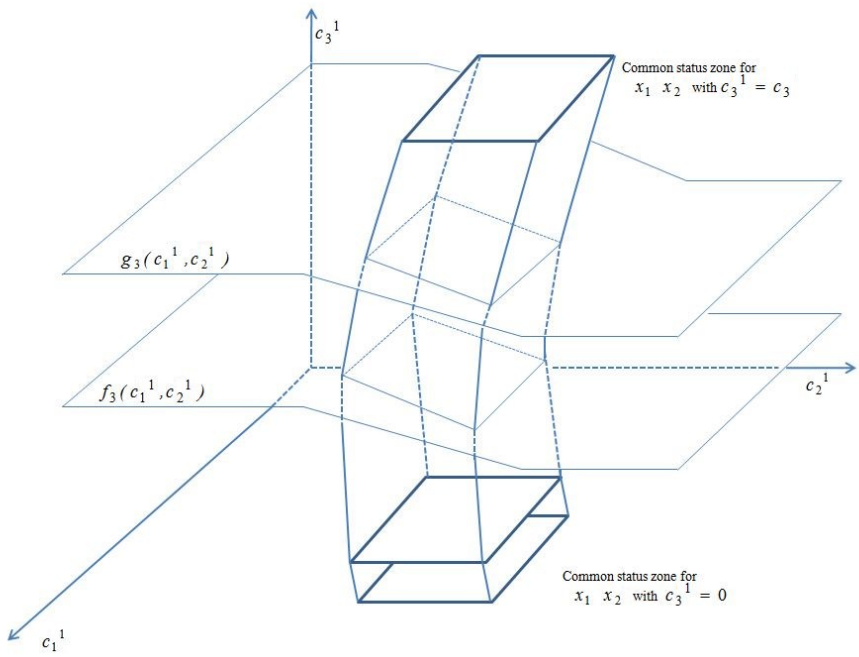

Figure 3.4: intersection of surfaces f_{n+1} and g_{n+1} and zone $U_{(1,\ldots,n)}$ in the case $n = 2$

Chapter 4

Two stage problems with three scenarios and more

4.1 Introduction

This chapter is divided into two parts : in the first part, na example of non TDI system with three scenarios is given, with extension to a more general class of non TDI illustrations. In the second part, a reduction is proposed, derivate from the set cover problem and from the reduction presented in [19].

4.2 A first example

In this section, we study the case where there exist more than two scenarios in the second stage. We will change our point of view by exhibiting a counter-example, where a fractional solution for (x, z) still compatible with all requirements leads to a higher value than for all integer vectors (x, z). We present a graph where there exist 3 first stage edges not directly connected, and 6 second stage edges.
For all first stage edges, the cost values are $c_1 = c_2 = c_3 = 5$.
In scenario $\mathscr{S}_1$, the cost function for second stage edges is $c_{11} = c_{21} = 6$ and $c_{31} = c_{41} = c_{51} = c_{61} = 0$.
In scenario $\mathscr{S}_2$, the cost function for second stage edges is $c_{32} = c_{42} = 6$ and $c_{12} = c_{22} = c_{52} = c_{62} = 0$.
In scenario $\mathscr{S}_3$, the cost function for second stage edges is $c_{53} = c_{63} = 6$ and $c_{13} = c_{23} = c_{33} = c_{43} = 0$.
There is no integer solution (x, z) where it is possible to take all second stage edges with positive strictly cost and strictly more than only one first stage edge, otherwise there would be a cycle (see figure 4.2). The cost of the edges is presented for every edge with a strictly positive cost in figure (4.2).

We can afford that best integer value is less or equal than $6 * 6 + 5 = 41$. Now

we propose to take $x_1 = x_2 = x_3 = \frac{1}{2}$ and, in the second stage, only edges with strictly positive costs: $z_{11} = z_{21} = 1$; $z_{32} = z_{42} = 1$; $z_{53} = z_{63} = 1$. This fractional solution gives a positive value of $6 * 6 + 3 * 5 * \frac{1}{2} = 43,5$. This clearly shows that this system is not TDI. All requirements are satisfied:

$$\begin{cases} \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, 3\}, \quad \forall S \subseteq E \\ (x, z_k) \in [0,1]^n \times [0,1]^q, k \in \{1, \ldots, 3\} \end{cases} \tag{4.1}$$

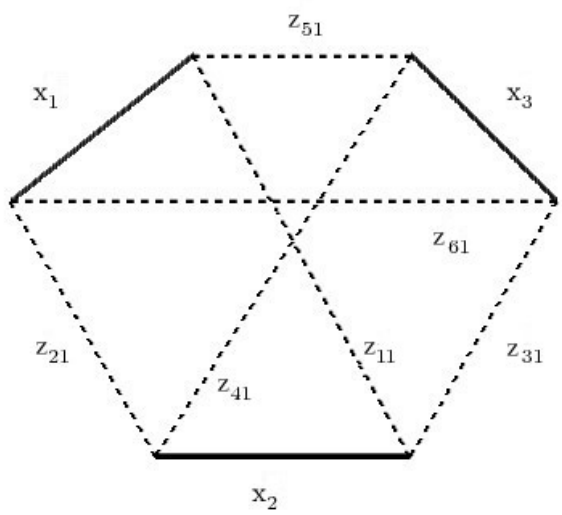

Figure 4.1: A first example : Complete graph for one scenario

4.3 A more general class of non TDI systems

When we began to study the question of TDIness for the multi-stage covering forest, it appeared that every simple example of graph treated in Porta gave integer optimal solutions. The first reason has been explained since when there are only two scenarios, the system is always TDI. The first example that shows that some formulation should not be TDI needed to behold at least 3 scenarios, and some first stage edges that where systematically covered by some second stage edges in one scenario and always picked up in another one. This kind of configuration is not very common indeed. Then, some first stage edge is never covered by second stage edges because of some structural reasons of geometry of the graph (see figure 4.3). We try to give some graphical characterization of non-TDI systems.

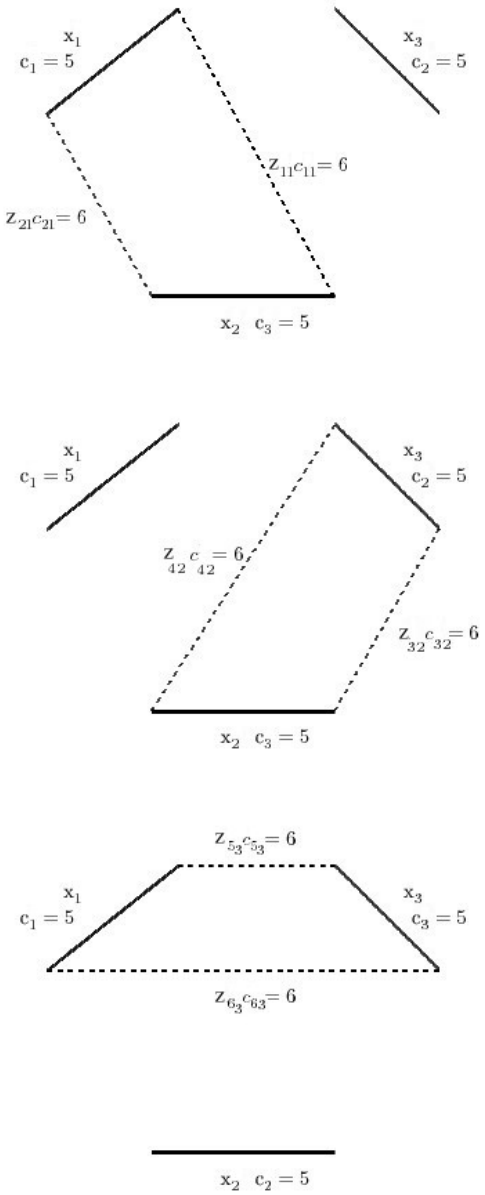

Figure 4.2: A first example : three scenarios with cost function

Theorem 31 *Suppose that there exist $K \geq 3$ scenarios and at least three first stage edges called x_1, x_2, x_3 such that there exist three scenarios k_1, k_2, k_3 with the following assumptions:*

1. *In scenario k_i for $i \in \{1, \ldots, 3\}$, x_i is isolated (connect a vertex of degree 1).*
2. *In scenario k_i, the two others first stage edges belong to a circuit (Definition 8) and are the only first stage edges of this circuit.*

Then the system is not TDI.

Proof. The proof is similar to the construction of the above example. Give the same value $c_1 = c_2 = c_3 = c$ to the three first stage edges, and in scenario k_i give some value $c' > c$ for any second stage edge in the circuit and 0 for all other second stage edges. Then, it is not possible to pick up every second stage edge in the three circuits and at least two first stage edges. This ensures that the fractional value where $x_1 = x_2 = x_3 = \frac{1}{2}$ has a better optimal value that any integer solution. ■

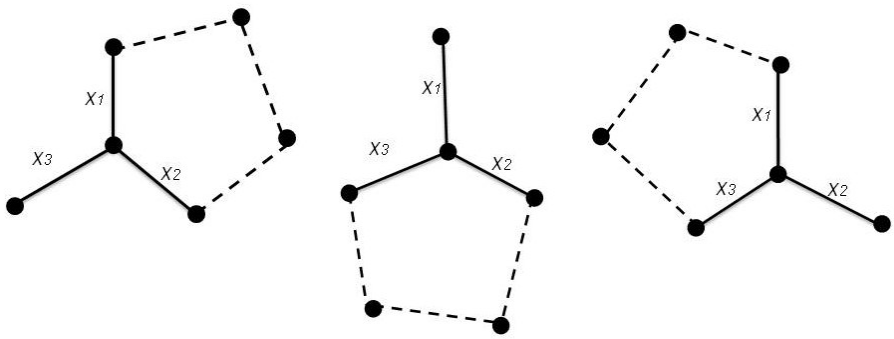

Figure 4.3: More general case : 3 scenarios with 3 circuits beholding two first stage edges and an isolated first stage edge in each scenario

In appendix, we give the formulations of these polyhedra in Porta.

In next section, we adapt the proof given by Flaxman et. al in [19] to analyze the reduction of our problem.

4.4 Reduction from set cover

Let $\mathscr{S} = \{S_1, S_2, \ldots, S_m\}$ be a set cover instance of V. The set cover problem is to find a minimum cardinality subcollection of $\mathscr{S}$ that covers all elements of V. We construct a SMaxST (Stochastic Maximum Spanning Tree) instance with

$n+m+1$ vertices by defining two stage cost functions. Denote the vertices by $\{r, v_1, v_2, \ldots, v_m, 1, 2, \ldots, n\}$. Set the first stage edges set as $\{\{r, v_1\}, \{r, v_2\}, \ldots, \{r, v_m\}\}$ and the cost function $c = 1$ for all first stage edges, set all remaining edges as second stage edges. Consider n scenarios for the second stage and define scenario $\mathscr{S}_j$ with the set $T_j = \{j\} \cup \{v_i : S_i \ni j\}$ and the cost function:

$$c(\{u,v\}) = \begin{cases} 2 & \text{if } \{u,v\} \in T_j^2 \quad \text{and } v \in \{v_1, v_2, \ldots, v_m\} \quad \text{and } u = j \\ 0 & \text{if } \{u,v\} \in T_j^2 \quad \text{and } \{u,v\} \in \{v_1, v_2, \ldots, v_m\}^2 \\ -\infty & \text{if } \{u,v\} \in (T_j, \overline{T_j}) \\ 2 & \text{if } \{u,v\} \in (\overline{T_j})^2 \quad \text{and } v \in \{v_1, v_2, \ldots, v_m\} \quad \text{and } u \in \{1, \ldots, n\} \\ 0 & \text{if } \{u,v\} \in (\overline{T_j})^2 \quad \text{and } \{u,v\} \in \{v_1, v_2, \ldots, v_m\}^2 \text{or } \{u,v\} \in \{1, \ldots, n\}^2 \\ 1 & \text{if } \{u,v\} \in (\overline{T_j})^2 \quad \text{and } v = r \quad \text{and } u \in \{1, \ldots, n\} \\ 0 & \text{if } \{u,v\} \in (\overline{T_j})^2 \quad \text{and } u = r \quad \text{and } v \in \{v_1, v_2, \ldots, v_m\}. \end{cases}$$

Note that this is the only part of this work where we analyze some graph with negative costs.

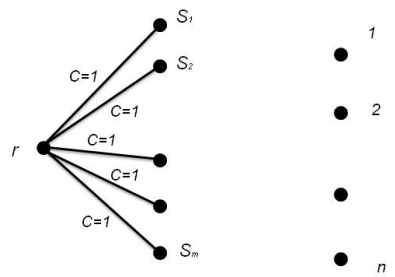

Figure 4.4: First stage edges with cost function

Select a value of p where $1 \leq p \leq m$. For $S_{i_1} \cup S_{i_2} \cup \ldots \cup S_{i_p}$ a minimal set cover selection, we choose at first stage the edges $\{r, v_{i_s}\}$ where $s = 1, \ldots, p$. When scenario j occurs, the tree can be completed by choosing for each $i \in \{1, \ldots, n\}$ with $j \neq i$ the edge $\{i, r\}$, and for any $v_i \notin T_j$ only one $\ell \neq j$ such that $\ell \in S_i$. For j, we select one specific $v_{i_j} \in T_j$ and choose $\{j, T_{i_j}\}$.

Now we prove that this choice is optimal. Suppose that the selected sets are not a covering set; there exists at least one edge $\ell \notin S_{i_1} \cup S_{i_2} \cup \ldots \cup S_{i_p}$. In scenario ℓ,

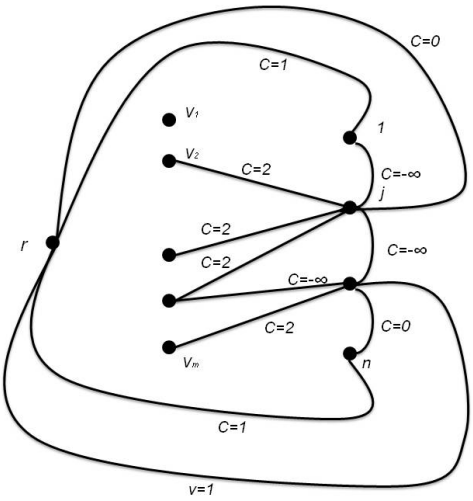

Figure 4.5: Second stage edges and cost for scenario j, the set T_j is isolated from the rest of the graph

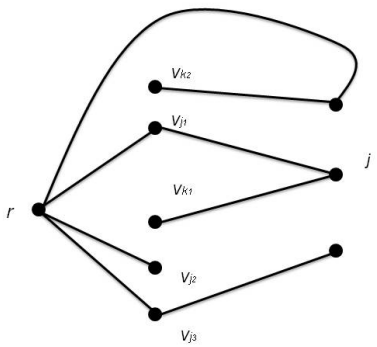

Figure 4.6: Maximal Spanning Tree when scenario j occurs. v_{j_2} is a corresponding vertex of the selected cover set S_{j_2} but $\{v_{j_2}, j\}$ cannot be selected in the maximal spanning tree since $\{r, v_{j_2}\}$ is selected during the first stage.

since the part T_ℓ is connected with infinitely negative costs with $\overline{T_\ell}$, any spanning tree is negative.

Chapter 5

Approximation in case of a non-TDI system

5.1 Introduction

Before commenting Stochastic Maximal Tree Problem, it is useful to describe Deterministic and Stochastic versions of two connected problems:

The Maximal Spanning Tree problem (MaxST) is very close to the Minimal Spanning Tree problem (MinST). These problems can be considered both in Deterministic version or Stochastic (2-stage) version. The Deterministic Maximal Spanning Tree Problem (DMaxST) is solved by the greedy algorithm in a polynomial time [40]. The complexity of the algorithm is merely the complexity of a sorting algorithm. The Deterministic Minimal Spanning Tree Problem (DMinST) is equivalent from a complexity point of view (replace the cost function c by c' where $c'(e) = c_{max} - c(e)$ for any edge e). However, this equivalence seems not to hold for Stochastic Maximal Spanning Tree (SMaxST) and Stochastic Minimal Spanning Tree (SMinST) problems [18].

Concerning the DMaxST problem, since two decades, numerous works have proposed to reduce the polynomial cost at the price of an approximation factor. Best randomized algorithms have reached a linear cost and are based upon a verification that a given solution is optimal [32]. In this approach, Karger et al [31] proposed a randomized algorithm with linear complexity for the DMinST problem.

There are two basic approaches with approximation algorithms in linear programming [58]: LP-rounding and primal-dual schemes.

LP rounding technics applied in multi-stage stochastic problems appear to be natural and bring efficient results.

Concerning the SMinST problem, first approaches have been presented in a restrictive frame of hypotheses corresponding to one or several items presented below:

- Choices made at the first stage build partial solutions that cannot be in-

validate at the second stage [17]. Escoffier Gourvès Monnot and Spanjaard qualify this property of "monotonicity" in [18].

- The number of scenarios is either an input of the problem [19] or not [18].
- There exists an inflation ratio which fixes the ratio of costs between first stage and second stage [27], or a global inflation factor which limits the same ratio [51].

The DMinST problem is connected with the Deterministic Set Cover Problem (DSCP) via reduction techniques displayed in section 4.4. We remind that the LP formulation of the DSCP has an integrality gap of 2. Meanwhile, Shmoys and Swamy [56] show that any ρ-approximation of the DSCP gives a 2ρ-approximation of the Stochastic Set Cover Problem (SSCP).

Unfortunately, these reduction techniques are not appropriate for the stochastic version of spanning trees because they induce an inflating number of scenarios. So the stochastic version SMinST is $O(\log n)$-hard to approximate by using these techniques. In the next section, we adapt the proof given by Flaxman et al in [19] to prove the reduction in our case.

Concerning the Stochastic Maximal Spanning Tree (SMaxST) problem, the authors in [18] show that this problem is **APX**-complete [1].

As a partial conclusion, Maximal or minimal stochastic spanning tree problems (SMST) appear to be hard to approximate. More precisely, [19] and [51] show that, via a reduction from set-cover, the SMinST problem is hard to approximate within a factor of min $\{\log n, \log k\}$, with a very general black-box model. And for SMaxST problem, Escoffier et al propose an approximation algorithm with guarantee $\frac{K}{2K-1}$ by improving techniques proposed by Kong and Schaeffer in [35]. We outline that in their model, every first stage edge is possibly a second stage edge, so that when K scenarios occur in their work, this corresponds to $K+1$ scenarios in our description. The results given in our first part concerning the case of only two scenarios is therefore not treated in their approach.

5.2 Different point of views in approximation

As mentionned in the introduction, mainly two techniques are possible to build approximation algorithms: LP rounding and Primal-Dual schemes. LP rounding is based at first step on the linearization of the problem, then, in a second step an algorithm decides to select or discard the edges whose frequency (or ponderate value) is greater than a given threshold. Primal-Dual method is more sophisticated and is usually the best way to design approximation algorithms with a guarantee of upper bound of OPT. The first step is the linearization of the problem in both approaches.

[1] A problem is **APX** if there exists a polynomial approximation algorithm with approximation ratio bounded by a constant.

5.3 An approximation algorithm in the case of more than two scenarios

The main idea in this section is to change the cost function c in order to turn the stochastic problem into a new one which gets an optimal integer solution. Since the greedy algorithm finds a solution for any scenario according to the attributed value of partial first stage edges, we progressively downsize the cost of first stages edges which are in conflict between several scenarios.

5.3.1 Introduction

We remind the two stage stochastic maximum problem formulation: For $E = X \cup Y$, X first stage edges and Y second stage edges, and for (c, c_k) weights vectors with $c \in \mathbb{R}^n$, $c_k \in \mathbb{R}^q, k = 1, \ldots, K$, consider the two stage stochastic maximum problem:

$$Z_{IP} = \begin{cases} max \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \sum_{j \in Y} c_{jk} z_{jk} \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_{jk} \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E \\ (x, z_k) \in \{0,1\}^n \times \{0,1\}^q, k \in \{1, \ldots, K\}. \end{cases} \tag{5.1}$$

Consider a split of the form

$$c_i = \sum_{k=1}^{K} c_i^k, \ i \in X$$

with

$$c_i^k \geq 0, \ i \in X$$

or in an equivalent vector formulation:

$$c = \sum_{k=1}^{K} c^k$$

with

$$c^k \geq 0$$

We proved in (25) that when for any given split, $(c^k, c_k), \ k = 1, \ldots, K$, if each first stage edge is either always or never picked simultaneously in every scenario by the greedy algorithm, then the system (5.1) is TDI.

5.3.2 Two new tools derived from the standalone scenario or deterministic problem

In a deterministic graph (or in a single scenario), the greedy algorithm leads to the construction of an increasing sequence of selected edges known as J_t. We suppose $c_1 \geq c_2 \geq \ldots \geq c_n$. The subset J_t is the partial subtree of free greedy edges built at step t. We call K_t the closure set of a partial greedy solution at step t. In the previous part of this work, we have seen that the status of one specific edge i depends on a specific threshold b. If $c_i \geq b$ then the edge will be chosen, while if $c_i \leq b$, the edge is dropped. We now extend this approach to the observation of the increasing sequence of subsets K_t. As we did before, we begin by considering that a new edge of cost c_e is theoretically introduced in a given graph. The status of this edge depends on a specific threshold b. We reformulate this question as the following one: is it possible to build some functions of real variable b taking values in the graph G?

We introduce two functions:

$$J : \begin{cases} \mathbb{R}^+ \to G \\ b \mapsto J_t \text{ where } t = sup\{index(c_t), c_t \geq b\} \end{cases} \quad \text{and } K(b) = sp(J(b)).$$

In other words, we describe the evolution of the progressive closure of the partial greedy solution as depending on a parameter b. b decreases from the heaviest weight down to zero. When b is strictly between two values of weights of the graph $c_t > b > c_{t+1}$, there is no change and the closure remains the same as the one gained for $b = c_t$. When b reaches any value c_t, $J(b)$ and $K(b)$ possibly change, depending on the fact that the examined edge is added or not to the greedy solution. The interesting point is that J can radically change when the weight vector c is slightly modified, but K has a relative stability that we study here.

Lemma 32 $K(b, c_1, c_2, \ldots, c_i \geq b, b > c_{i+1}, \ldots, c_n) = K(b, c_1, c_2, \ldots, c_i, 0, \ldots, 0)$ *where the weights of edges lighter than* b *are replaced by* 0.

Proof. This lemma is clearly a consequence of the definition of $J(b)$. Since in $J(b)$, the only edges examined are those with a weight greater than b, then $J(b, c_1, c_2, \ldots, c_i \geq b, b > c_{i+1}, \ldots, c_n) = J(b, c_1, c_2, \ldots, c_i, 0, \ldots, 0)$. The closure building of K from J does not depend on the weights of the graph, so the equality is true for K too. ■

We still need some extended version of the greedy algorithm: Consider a graph $G = (V, E)$ where some specific edges have been definitely selected to belong to any solution of any covering forest. Let say that these edges form a set called $S_0 = \{\tilde{x_1}, \ldots, \tilde{x_\alpha}\}$ of cardinality α and that S_0 contains no cycle. With these assumptions, the rank of S_0 is equal to α. Now we consider the problem of finding the maximal independent set F containing S_0 with maximal cost $c(F)$. We index

the edges not in S_0 by $x_1, \ldots, x_n$, for any subset S in E we define $r_0(S) = r(S \cap S_0)$ We formulate this problem this way:

$$\mathrm{Z(S_0)}_{IP} = \begin{cases} max \sum_{j=1}^{n} c_j x_j \\ x \in \{0,1\}^n \\ \sum_{j \in S} x_j \leq r(S) - r_0(S) \quad \forall S \subseteq E. \end{cases} \tag{5.2}$$

The dual problem associated with z_{IP} is :

$$\mathrm{Z(S_0)}_{LD} = \begin{cases} min \sum_{S \subseteq E} (r - r_0)(S) y_S. \\ \sum_{S: j \in S} y_S \geq c_j \quad \forall j \in \{1, \ldots, n\} \\ y_S \geq 0 \quad \forall S \subset E \end{cases} \tag{5.3}$$

Theorem 33 *The system 5.3.2 is TDI, the greedy algorithm gives the optimal solution.*

Proof. The algorithm is exactly the same as in 2.2.2. We call $\mathscr{F}$ the set of independent subsets in E. Rank the elements of E who are not in S_0 so that $c_1 \geq c_2 \geq \ldots \geq c_n$.

1. Let us begin with $J_0 = S_0$, $t = 1$.

2. Iteration t : If $c_t \leq 0$ then stop and $S^G = J_{t-1}$.

3. If $c_t > 0$ and $J_{t-1} \cup \{t\} \in \mathscr{F}$, then set $J_t = J_{t-1} \cup \{t\}$.
   If $c_t > 0$ and $S_{t-1} \cup \{t\} \notin \mathscr{F}$ then set $J_t = J_{t-1}$.

4. If $t = n$ stop and $S^G = J_t$

5. Set t to $t + 1$

The greedy solution is $\{x_i : i \in S^G\} \cup S_0$.

We call p the number of chosen edges and $S^G = J_p$.

Denote $J_\tau = \{i_1, \ldots, i_\tau\}, 1 \leq \tau \leq t$ and $\kappa_\tau = sp(J_\tau)$. The selected sets in the dual formulation are the sets κ_τ, and the dual variables are fixed like before:

$y_{\kappa_t} = c_{j_t} - c_{j_{t+1}} \quad for\, t = 1, \ldots, p-1$
$y_{\kappa_p} = c_{j_p}$
$y_S = 0 \quad otherwise.$

Like previously in the greedy algorithm, we easily check that for every $j \in \{1, \ldots, n\} : \sum_{S/j \in S} y_S \geq c_j$.

Then, we compute the sum : $\sum_{S \subseteq E} (r - r_0)(S) y_S = \sum_{t=\alpha+1}^{\alpha+p-1} (t - r_0(\kappa_t))(c_{j_t} - c_{j_{t+1}}) + (p + \alpha - r_0(\kappa_p)) c_{j_p}$
Since for every κ_t, we have $r(\kappa_t) = t + r_0(S_0) = t - \alpha$, we have the same sum as previously, and the same re-indexation gives: $\sum_{S \subseteq E} r(S) y_S = \sum_{t=1}^{p} c_{j_t}$.
This shows that $z(S_0)_{IP}$ and $z(S_0)_{LD}$ have the same objective value. ■

5.3.3 Application to the multi-stage problem

We notice $K_k(b, (c^k, c_k))$ the closure described above specifically in scenario k. Reminding that every first stage edge cost is split according to $c \mapsto \prod (c^k, c_k)$. In this notation, upper index is for a first stage edge, and lower index is for second stage index. For instance c_1^3 is the partial value of the first stage edge in scenario 3, and c_{45} is the value of the fourth second stage edge in scenario 5.

Now suppose that (5.1) is not TDI. That means that the resolution of (5.1) gives some fractional solution.

Lemma 34 *The subset of variables whose values are fractional always contains first stage edges.*

Proof. Suppose that in scenario k, there is no fractional first stage edge and at least one fractional second stage edge. Pick up the first stage edges selected in the optimal solution and call it S_0, then consider only the graph associated to scenario k, the second stage edges as remaining edges possibly chosen $x_{1k}, \ldots, x_{nk}$ and definitely let the other first stage edges unchosen in this graph. The optimal solution with fractional second stage edges has to be compared with the one encountered in this new formulation of the type of 5.3.2. The theorem 33 proves that a solution

with no fractional second stage in scenario k is better than the optimal solution. ■

Lemma 35 *Any fractional solution involves at least two first stage edges*

Proof. Suppose there is only one first stage fractional edge in the optimal solution and possibly several complete first stage edges. Pick up the complete first stage edges selected in S_0 and consider the graph with only the remaining fractional first stage edge as a single first stage edge. According to theorem 28, the associated system is TDI and ensures that the optimal solution has not its only first stage edge fractional. ■

Lemma 36 *In case of a fractional optimal solution, any split of first stage edge costs never gives a uniform status to fractional edges. That means that there always exists a scenario where an edge is covered and another where it is selected when considering separate greedy algorithm for any given split.*

Proof. This is a consequence of theorem 26. ■

Lemma 37 *Any fractional solution beholds at least 3 fractional edges.*

Proof. Suppose there are only two fractional edges. According to lemma 35, these edges are first stage edges. So there is no more fractional edge in second stage. Pick up the complete selected first stage edges in S_0 and keep the fractional ones as new first stage edges. We are now in the case of only two first stage edges and K scenarios examined in 3.3.2. The idea is to reduce the complexity down to the case of two scenarios only. Consider any split along K scenarios for the two fractional first stage edges and examine only two specific scenarios j and i where these edges have an opposite status. Balance the partial split of these two edges between scenarios i and j without changing partial costs along the other scenarios. Theorem 29 ensures that there is a correct balance giving the same status that can be chosen indifferently as chosen or covered. Choose the status corresponding to the scenarios where the given split gives no conflict between the two edges.

■

Some extension of this lemma leads to the following result:

Lemma 38 *Any fractional solution involves at least three first stage edges.*

5.3.4 Approximation algorithm

Suppose that the problem 5.1 is not TDI. That means that there exists some specific cost vector c, for which with any split of c, some first stage edge x_i cannot get the same status in every scenario. In that case, $Z_{IP}(c) < Z_{LD}(c)$. We propose to decrease the weight c_i of the fractional first stage edge down to zero one by one until we reach some equality $Z_{IP}(c') = Z_{LD}(c')$. This leads to set every conflicting edge to a status of covered in every scenario.

1. Solve with polynomial complexity method the problem (5.1).
2. In case of fractional solution, select one first stage edge with fractional value of highest weight and drop its weight to zero. This leads to solve the problem with a new vector cost c'.
3. Solve the new linear problem and observe if its solution is fractional or not. In case of a fractional solution, iterate process to point 2. In case of an integer solution jump to point 4.
4. Select the remaining set of complete first stage edges as S_0 and for every scenario j, solve with the greedy algorithm the system formulated by 5.3.2 with second stage edges only as decision variables. In every scenario, this optimal solution is called $Z(S_0)_{IP,j}$.

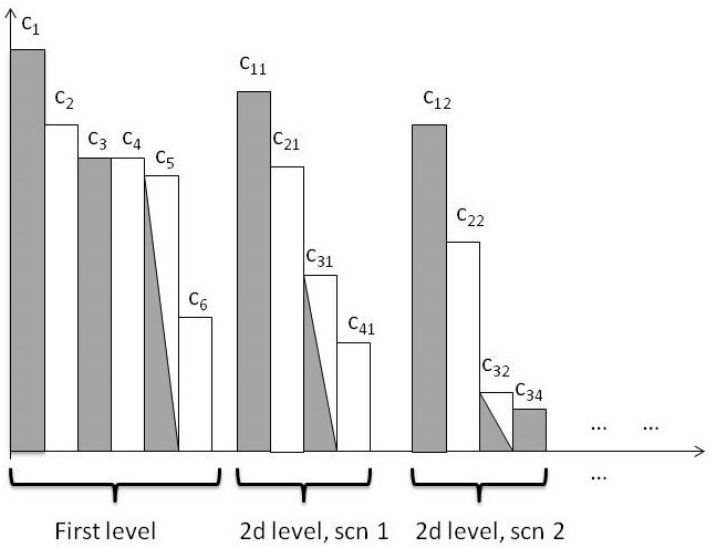

Figure 5.1: Illustration of step 1: the optimal solution of 5.1 is not TDI (first stage edge $n°5$ has a fractional value), while edge $n°1$ is selected (grey color) and edge $n°2$ is not selected (white color). Among second stage edges, edges $n°3,1$ and $n°3,2$ have a fractional value.

5.3.5 Evaluation of the approximation

Note that the number of iterations is bounded by the number of first stage edges since every iteration removes one edge. This construction leads to a set of complete

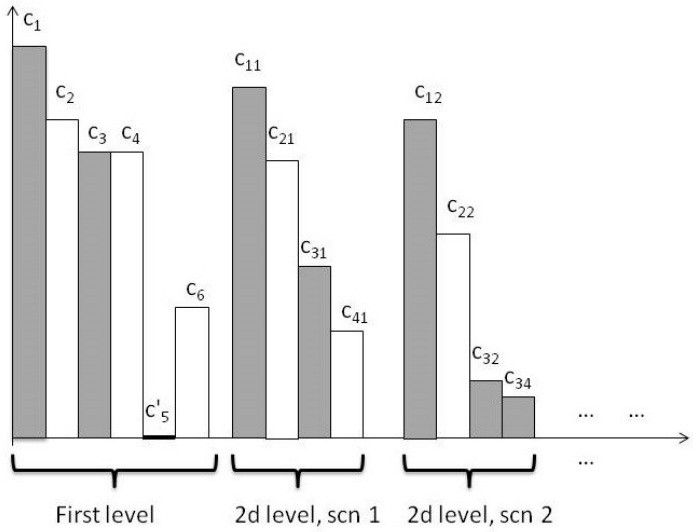

Figure 5.2: Illustration of step 2: Reduce the value of c_5 to $c'_5 = 0$, then some changes occur in the optimal solution and for instance, edges $n°3, 1$ and $n°3, 2$ are now completely selected

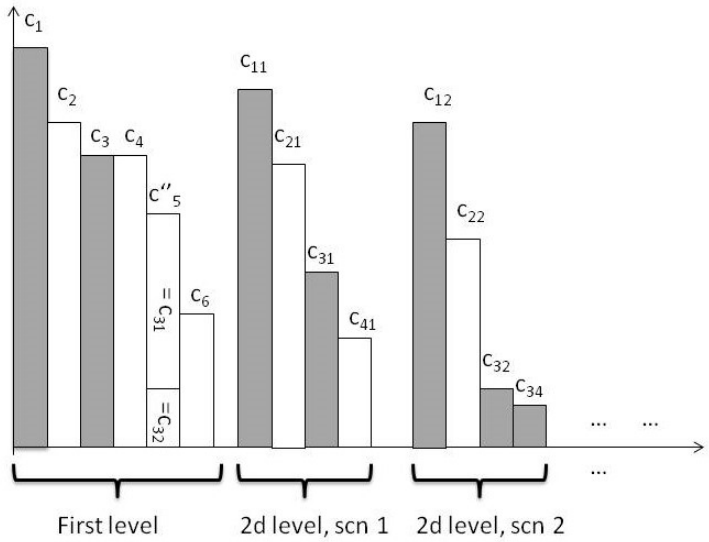

Figure 5.3: Illustration of step 4: Observe in every scenario the graphical status of x_5, determine the associated threshold for which the status change in the local greedy algorithm. Lift up c'_5 up to this value without changing the status. The total amount for c_5 that still gives a TDI system is $c_{31} + c_{32}$

integer values for first and second stage variables. The objective value of this solution is $Z_{app} = c(S_0) + \sum_{j=1}^{k} Z(S_0)_{IP,j}$. Obviously, this approximation is a lower bound of Z_{IP} of problem 5.1. Denote by x_f the fractional optimal value associated to Z_{LP} (2.7), by x_I the integer optimal value associated to Z_{IP} (6.2) (which is unknown) and by x'_I the integer optimal value associated to $Z_{app} = Z_{IP}(c')$. We get the following inequalities:

$$\begin{aligned} cx_f &\geq cx_I \\ cx_f &\geq c'x_f \\ cx_I &\geq cx'_I \\ cx'_I &\geq c'x'_I \\ c'x'_I &\geq c'x_f. \end{aligned}$$

These inequalities lead to

$$(c - c')x_f \geq cx_I - cx'_I \geq 0.$$

5.3.6 Illustration

The figure 5.4 illustrates a graph with three scenarios. Despite the fact that in every scenario, the set of second stage is the same, in the figure, second stages with null cost are not shown. The indexes in brackets are those used in PORTA to check possible fractional values. PORTA gives all possible summits (5 fractional summits among 728 for the system) See annex A for inequalities. In scenario 1, the first stage edge x_3 belongs to a triangle with the edges x_{31} and x_{41}. The cost vector is :

$$\begin{pmatrix} c_1 & c_2 & c_3 & c_{11} & c_{21} & c_{31} & c_{41} & c_{12} & c_{13} & c_{23} \\ 5 & 5 & 5 & 10 & 10 & 10 & 4 & 10 & 10 & 10 \end{pmatrix}.$$

This cost vector shows a fractional optimal value $\frac{139}{2}$ (x_1,x_2 and x_3 have a fractional value). The algorithm leads to choose one first stage edge with fractional solution (x_3) and to decrease c_3 to 0. Then the new optimal value is 69 with an integer solution :

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_{11} & x_{21} & x_{31} & x_{41} & x_{12} & x_{13} & x_{23} \\ 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

In scenario 1, x_3 is covered in the greedy algorithm by x_{41} whose cost is 4. Reducing finally the cost from 5 to 4 for c_3, we get an integer optimal solution with an optimal value of 69. In this situation, the approximation is bounded by $(c - c')x_f = (5-4)\frac{1}{2} = \frac{1}{2}$.

In annex B, we present some computations with Maple 16 associated to this graph and this algorithm.

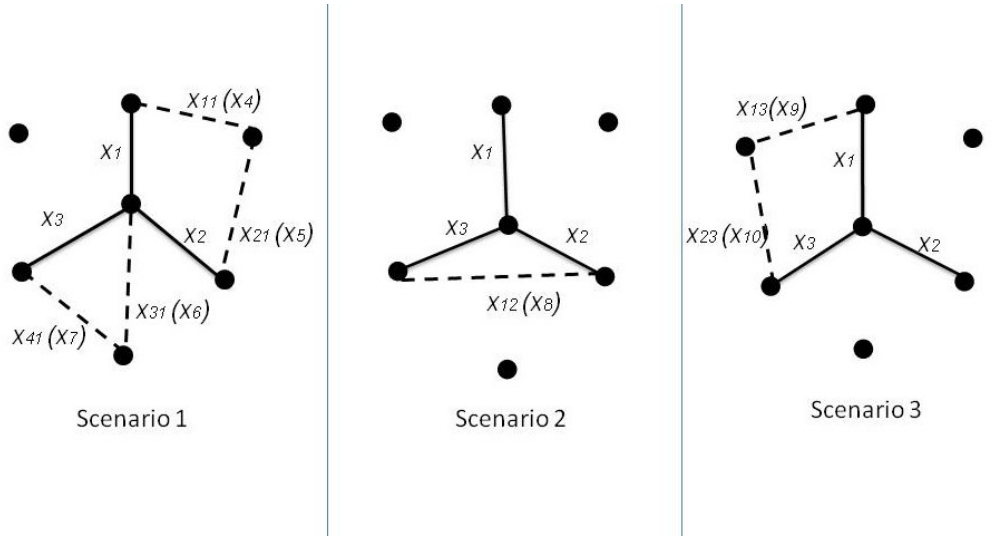

Figure 5.4: A graph with 3 scenarios

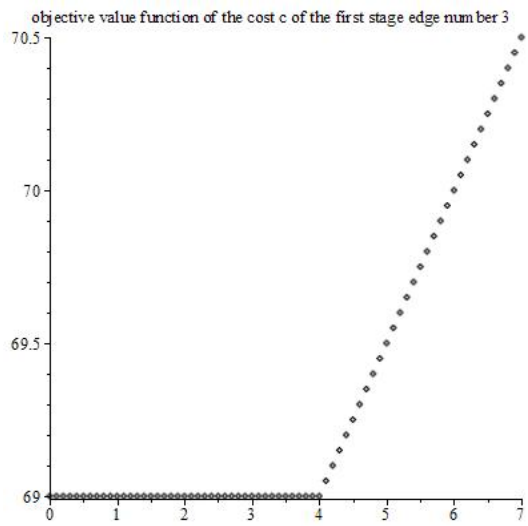

Figure 5.5: Objective value function of the cost of x_3, the threshold $c_3 = 4$ is the value where x_3 changes of status.

Chapter 6

Exploring inequalities in the stochastic maximum weight forest

6.1 Introduction

Let $G = (V, E)$ be a non-oriented graph. Let $V' \subset V$ be a subset of the vertices, we call $G(V')$ the subset of edges e whose two extremities are in V'. The covering forest of G are the subgraphs of maximal cardinality satisfying all the following constraints:

$$\forall V' \subset E, |\{e \subset G(V')\}| \leq |V'| - 1.$$

These constraints are easy to check, but, due to the arbitrary choice of a subset $V' \subset V$, the number of constraints is exponential with the size of the graph. The purpose of this section is to explain how it is possible to formulate an equivalent set of constraints with a polynomial number of inequalities.
It is possible to check if a subgraph V' contains a cycle in a polynomial time. Any graph with no cycle has at least two vertices of degree 1. The following algorithm checks in a polynomial time if the graph is acyclic:

1. Set $i = 1$ and $F_1 = V'$

2. remove from F_i the vertices of degree 1, call F_{i+1} the remaining set of vertices, $i \mapsto i + 1$

3. if $F_n \neq \emptyset$ then V' has a cycle, else V' has no cycle.

Unfortunately, this verification is not easy to translate into inequalities. In [43], the authors propose a polynomial formulation, in terms of inequalities. This work is based upon [45] for the spanning tree polytope by a straightforward adaptation for non-complete graphs. The first step of this work needs the graph be orientated.

6.2 Polynomial Formulation

6.2.1 Orientation of a graph

The following results are given for a deterministic graph. Since the stochastic formulation of the SMWF leads to handle separate variables for each scenario, it is possible to extend these results to a two-stage formulation.

Theorem 39 *Let $G = (V, E)$ be a graph, and F a subgraph. F is acyclic $\iff$ for any node $k \in F$, there exists an orientation F_k such that :*

1. *There is no entering arc on the referential node k*
2. *There is at most one entering arc on each node $j \neq k$.*

Proof. Suppose that F is acyclic. Consider a node k and organize the graph into a pending tree for all the elements of the connected component of k in F. For the other nodes of F in different connected components, choose an arbitrary node $k_2, \ldots, k_p$ and consider the corresponding pending trees. For all $j \in F$ we define the distance to the referential node k_i of its connected component as the minimal number of nodes to pass to reach j from k_i in F. We orientate each edge in F according to the growing sequence of the nodes linked. In an acyclic forest, each node can only be reached in a single way. This orientation clearly satisfies assumptions of the theorem.
Conversely, suppose there is a cycle in F, choose a specific node k of the cycle. It is clear that it is not possible to satisfy both conditions for the arcs belonging to the cycle. ■

6.2.2 Polynomial Formulation

We recall the stochastic formulation of the SMWF problem :

$$z_{IP} = \begin{cases} max \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \pi_k \sum_{j \in Y} c_j(k) z_j^k \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z_j^k \leq r(S), \quad k \in \{1, \ldots, K\}, \quad \forall S \subseteq E \\ (x, z^k) \in \{0,1\}^n \times \{0,1\}^q, k \in \{1, \ldots, K\}. \end{cases} \tag{6.1}$$

We add the modeling of the choice of a referential node and the orientation of the pending graph : define for every node v in V chosen as referential node, and for every scenario k and for every edge x_j, two binary variables λ_{vj1}^k and λ_{vj2}^k, for z_j, two binary variables μ_{vj1}^k and μ_{vj2}^k. These binary variables are defined according to the following rules:

1. λ^k_{vj1} is associated to an oriented sense between the two nodes connected by x_j. If (u, w) are the extremities of the edge, we note $\lambda^k_{vj1} \to u$ if the orientation associated to λ^k_{vj1} is pointing on u, and $\lambda^k_{vj2} \to u$ on the opposite case.

2. λ^k_{vj2} is associated to the opposite sense.

3. λ^k_{vj1} is set to 1 when for referential node v, the oriented sense of the arc corresponds to the abstract orientation of extremities, and is set to 0 otherwise.

4. equivalent formulation for μ^k_{vj1} and μ^k_{vj2}

The formulation of the SMWF is now :

$$z_{IP} = \begin{cases} max \sum_{j \in X} c_j x_j + \sum_{k=1}^{k=K} \pi_k \sum_{j \in Y} c_j(k) z^k_j \\ \sum_{j \in S \cap X} x_j + \sum_{j \in S \cap Y} z^k_j \leq r(S), \quad \forall S \subseteq E \\ \lambda^k_{vj1} + \lambda^k_{vj2} = x_j, \forall v \in V, \forall j \in \{1, \ldots, n\}, \\ \mu^k_{vj1} + \mu^k_{vj2} = z^k_j, \forall v \in V, \forall j \in \{1, \ldots, q\}, \\ \sum_{j: \lambda^k_{vj1} \to u \ or \lambda^k_{vj2} \to u} \lambda^k_{vj1} + \lambda^k_{vj2} \leq 1 \quad \forall u \neq v \\ \sum_{j: \lambda^k_{vj1} \to v \ or \lambda^k_{vj2} \to v} \lambda^k_{vj1} + \lambda^k_{vj2} = 0 \\ \sum_{j: \mu^k_{vj1} \to u \ or \mu^k_{vj2} \to u} \mu^k_{vj1} + \mu^k_{vj2} \leq 1 \quad \forall u \neq v \\ \sum_{j: \mu^k_{vj1} \to v \ or \mu^k_{vj2} \to v} \mu^k_{vj1} + \mu^k_{vj2} = 0 \\ (x, z^k) \in \{0,1\}^n \times \{0,1\}^q, \\ (\lambda^k_{vj1}) \in \{0,1\}^{nK}, (\lambda^k_{vj2}) \in \{0,1\}^{nK} \\ (\mu^k_{vj1}) \in \{0,1\}^{qK}, (\mu^k_{vj2}) \in \{0,1\}^{qK} \\ k \in \{1, \ldots, K\}. \end{cases} \tag{6.2}$$

The equations $\sum_{j: \lambda^k_{vj1} \to u \ or \lambda^k_{vj2} \to u} \lambda^k_{vj1} + \lambda^k_{vj2} \leq 1 \quad \forall u \neq v$ stand for the condition that there is at most only one entering arc on every node (and *simile* for the equations in μ). The equations $\sum_{j: \lambda^k_{vj1} \to v \ or \lambda^k_{vj2} \to v} \lambda^k_{vj1} + \lambda^k_{vj2} = 0$ stand for no entering arc on the referring node. The model presented here is a slight simplification of the model developed in [43]. With this compact extended formulation we can now handle instances with up to 40 nodes.

Inst.	Instance parameters				(RP_1)		(RP_2)	
	$\|V\|$	N	M	P	# Constr.	# Var.	# Constr.	# Var.
1	5	5	5	5	130	30	675	550
2	5	5	5	50	1300	255	6750	5500
3	5	5	5	100	2600	505	13500	11000
4	8	14	14	5	1235	84	2730	2380
5	8	14	14	50	12350	714	27300	23800
6	8	14	14	100	24700	1414	54600	47600
7	10	22	23	5	5065	137	5285	4725
8	10	22	23	50	50650	1172	52850	47250
9	10	22	23	100	101300	2322	105700	94500
10	12	33	33	5	20415	198	9075	8250
11	12	33	33	50	204150	1683	90750	82500
12	12	33	33	100	408300	3333	181500	165000
13	15	52	53	5	163760	317	17585	16275
14	15	52	53	50	1637600	2702	175850	162750
15	15	52	53	100	3275200	5352	351700	325500
16	20	95	95	5	5242775	570	41325	38950
17	20	95	95	10	10485550	1045	82650	77900
18	20	95	95	25	26213875	2470	206625	194750
19	30	217	218	5	5368708965	1307	138110	132675
20	30	217	218	10	10737417930	2397	276220	265350
21	40	390	390	5	5497558138675	2340	325650	315900
22	6	3	6	3	171	21	684	585
23	8	4	10	4	988	44	2144	1904
24	8	4	11	5	1235	59	2680	2380
25	14	6	30	4	65476	126	11308	10556

Table 6.1: Dimensions of the instances.

6.3 Computational experiments

Numerical experiments have been carried out on a Pentium IV, 1 GHz with 2G-RAM under windows XP. The source codes are generated with Matlab and the optimization models are solved by CPLEX 12. We report preliminary results for randomly (1–21) and arbitrarily (22–25) generated instances.

The dimensions of the instances are in the Table 1. (RP_1) and (RP_2) correspond to the linear relaxations of the models (P_1) and (P_2), respectively. The first column identifies the instance number. Columns 2-5 provide the parameters $|V|$, $N = |E_D|$, $M = |E_S|$ and $P = |S|$. Columns 6-7 and 8-9 show the number of constraints and variables for (RP_1) and (RP_2), respectively. For the model (RP_1), the number of variables is of the order of $\mathcal{O}(N + MP)$ and the number of constraints, $\mathcal{O}(2^{|V|}P)$. For the model (P_2), the number of variables and constraints are of the order of $\mathcal{O}(P|V|^3)$. In the first part of Table 2 we report numerical results for random generated instances. Column *Inst.* identifies the instance from the Table 1. Columns 2-3, 4-5, 6-7 and 8-9 give the optimal solution value and the cpu time in seconds for (P_1), (RP_1), (P_2) and (RP_2), respectively. We have that the optimal relaxed solutions are all integer. For the random generated instances, the data for columns 6 and 7 are equal to those for columns 8 and 9, respectively. This means that these random generated instances have been proved easy. The

Inst.	(P_1)	cpu(s)	(RP_1)	cpu(s)	(P_2)	cpu(s)	(RP_2)	cpu(s)
Random generated instances								
1	163.0000	0.34	163.0000	0.32	163.0000	0.36	163.0000	0.36
2	178.6012	0.36	178.6012	0.36	178.6012	1.50	178.6012	1.50
3	176.4886	0.41	176.4886	0.41	176.4886	2.94	176.4886	2.94
4	460.0961	0.42	460.0961	0.39	460.0961	0.53	460.0961	0.53
5	395.5182	0.81	395.5182	0.75	395.5182	3.98	395.5182	3.98
6	460.7503	3.38	460.7503	1.36	460.7503	12.91	460.7503	12.91
7	585.3537	0.73	585.3537	0.69	585.3537	0.78	585.3537	0.78
8	589.5790	5.22	589.5790	2.66	589.5790	6.81	589.5790	6.81
9	612.3388	14.06	612.3388	5.50	612.3388	27.20	612.3388	27.20
10	808.9535	2.97	808.9535	2.36	808.9535	0.92	808.9535	0.92
11	809.6658	44.58	809.6658	19.00	809.6658	28.12	809.6658	28.12
12	855.6202	107.17	855.6202	37.00	855.6202	57.48	855.6202	57.48
13	1182.2429	44.23	1182.2429	27.05	1182.2429	2.83	1182.2429	2.83
14	-	-	-	-	1142.6433	161.94	1142.6433	161.94
15	-	-	-	-	1108.7019	1025.52	1108.7019	1025.52
16	-	-	-	-	1616.3587	15.67	1616.3587	15.67
17	-	-	-	-	1586.4263	37.13	1586.4263	37.13
18	-	-	-	-	1579.3486	759.53	1579.3486	759.53
19	-	-	-	-	2537.2853	254.66	2537.2853	254.66
20	-	-	-	-	2649.1216	3425.19	2649.1216	3425.19
21	-	-	-	-	3540.0685	4682.42	3540.0685	4682.42
Arbitrarily generated instances								
22	41	1.484	43.500	0.422	41	0.011	43.500	0.010
23	94	0.515	97.333	0.391	94	0.028	97.333	0.006
24	118	0.406	120.500	0.391	118	0.032	120.500	0.006
25	202	14.375	204.500	11.563	202	0.110	204.500	0.044

"-": No solution found due to CPLEX shortage memory.

Table 6.2: Numerical results for random and arbitrarily generated instances.

model (P_1) is very limited in solving those instances with more than 13 nodes. However, generally solving their corresponding linear relaxations (RP_1) takes less cpu time than by using the linear relaxation (RP_2) for these instances. Note that CPLEX finds integer solutions for (RP_1) in smaller cpu time than for (P_1) for these instances. In the second part of Table 2 we report numerical results for the arbitrarily generated instances of the Table 1. These instances intend to show the difficulty of the problem. These results give some insights about the complexity of this problem (this is an open question). Note that despite the reduced dimensions of these instances, and the fact that we do not have many instances to perform exhaustive numerical experiments, the new compact model obtains all their optimal solutions in considerable fewer cpu time than the model (P_1).

6.4 Conclusion

In this chapter, we proposed a polynomial size formulation for the stochastic maximum weight forest problem. This formulation is based on the one for the spanning tree polytope of complete undirected graphs [46]. We extend the model in [46] to deal with forests in non-complete graphs. For this, we proposed a new theorem characterizing forests in undirected graphs, which is important to prove the correctness of the new model.

Part II

Continuous Stochastic modeling

Chapter 7

The Stochastic Knapsack Problem

7.1 Introduction

The deterministic knapsack problem is a well known and well studied NP-hard combinatorial optimization problem. It consists in filling a knapsack with items out of a given set such that the weight capacity of the knapsack is respected and the total reward maximized. There exist several variations of the standard problem, as the multi-objective knapsack problem, or the bounded or unbounded knapsack problem. The connection with other NP-hard problems is given in [23]. In the deterministic problem, all parameters (item weights, rewards, knapsack capacity) are known. In the stochastic counterpart, some of these parameters are assumed to be random, i.e. not known at the moment the decision has to be made.

We consider a stochastic knapsack problem of the following form : A set of n items with the following characteristics:

- item i has a weight following of random variable χ_i, independent and normally distributed with mean $\mu_i > 0$ and standard deviation σ_i.
- the reward of i is equal to the weight. The choice of a reward per weight unit can be motivated by the fact that the value of an item often depends on the quantity of this item.

The choice of normal distribution was motivated by computational developments. All the methods used and results presented in this work are still valid in the case where the rewards are deterministic or random but independent of the weights. The fact that the rewards per item are normally distributed is used in some aspects of the proofs, but should be extended for any other distribution with similar properties. We denote by χ, μ, σ and r the corresponding n-dimensional vectors. The objective is to maximize the expected total gain $\mathbb{E}[\sum_{i=1}^{n} r_i \chi_i x_i]$. The knapsack problem has a fixed weight capacity $c > 0$. The formulation is called chance constrained optimization or probabilistic constraint optimization problem :

Chance Constrained Knapsack Problem ($CCKP$)

$$\max_{x\in\{0,1\}^n} \mathbb{E}\left[\sum_{i=1}^{n} r_i\chi_i x_i\right] \tag{7.1}$$

$$\text{s.t.} \qquad P\{g(x,\chi)\leq c\}\geq p \tag{7.2}$$

where $\mathbb{E}[\cdot]$ denotes the expected value, $g(x,\chi)=\sum_{i=1}^{n}\chi_i x_i$ is the total weight of the chosen items, $p\in(0.5,1]$ is the prescribed probability.
The choice of p is a decision parameter that restricts the percentage of cases where the capacity is exceeded. p is not connected with the amount of overweight.

The choice of the formulation as a chance constrained problem will be discussed in the next subsection. Many other approaches have been held in the stochastic knapsack problem. Another formulation is the stochastic knapsack with simple recourse, where a penalty is calculated with a factor d when the total weight exceeds the capacity of the knapsack :

$$\max_{x\in\{0,1\}^n} \mathbb{E}\left[\sum_{i=1}^{n} r_i\chi_i x_i\right] - d\mathbb{E}[[g(x,\chi)-c]^+] \tag{7.3}$$

This classical formulation has been studied in [1],[11],[33] and recently in [47]. We still mention two other important issues concerning stochastic knapsack problems, but more far from our approach : first, dynamic decision problems - where some information concerning the items are supposed to be known during the process [42], [6]- and where a decision policy has to be established, and, secondly, approximation algorithms [26], [34].

The results presented in this part of the thesis belong to a larger work on stochastic knapsack problems [39] [38] where different methods are analyzed, from branch and bound to stochastic gradient algorithms, and presented numerical results. Due to its combinatorial nature, $CCKP$ can be treated by using a branch-and-bound framework as presented in [39]. To obtain upper bounds, the authors propose to solve there the corresponding continuous optimization problem. Then, the relaxed formulation of this optimization problem is treated by using a stochastic gradient type algorithm. Several technical and theoretical difficulties are raised in these methods. This part of the thesis focuses on these difficulties and the way they are treated. Mainly we distinguish difficulties connected to the reformulation of the constraint with smooth functions, and those connected to the convergence of the stochastic algorithm. Stochastic gradient based methods have been applied to both combinatorial formulation and continuous formulation of the stochastic knapsack. We will shortly present some numerical results for comparison with those given in [38].

7.2 Global frame for the Chance constrained formulation

The problem 7.1 belongs to a wide class of formulations of stochastic optimization problems called optimization problems withs chance constraints. Chapter 4 of [55] gives an overview of these formulations. Chance constraint optimizations are used when it seems necessary to control the risk. The first part of this thesis was an example of a model with two-stage formulation. In multi-stage formulation, uncertainty is taken into account by a decision process during realization of the events. The multistage model leads naturally to the on-line decision modeling, where decisions occur during the execution of the process. Chance constrained optimization is rather different from multi-stage modeling. A typical example of chance constrained modeling is the Value at Risk constraint : optimization has to be made under the assumption that the loss will not exceed a given threshold according to a good ratio of probability. A basic example is given by n investment opportunities with random return rates $R_1, \ldots, R_n$ and an initial capital c to invest in order to maximize the outcomes, under the condition that the chance of loosing more than a fixed percentage η of the capital will not exceed a probability p. The formulation is

$$\max_{x \in [0,c]^n} \mathbb{E}\left[\sum_{i=1}^{n} R_i x_i\right] \tag{7.4}$$

$$\text{s.t.} \qquad P\{\sum_{i=1}^{n} R_i x_i \geq c\eta\} \geq p \tag{7.5}$$

$$\sum_{i=1}^{n} x_i = c. \tag{7.6}$$

A wide covering formulation of these problems is given by :

$$\max f(x) \tag{7.7}$$

$$\text{s.t.} \qquad P\{g_j(x,\chi) \leq 0, j \in \mathbb{J}\} \geq p \tag{7.8}$$

$$x \in \mathbb{X}, \tag{7.9}$$

where $\mathbb{X} \subset \mathbb{R}^n$ is non-empty, $f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}$, $\mathbb{J}$ is a set of indexes (here, there is only one constraint), and χ is a s-dimensional random vector. The function f can itself be an expected function. Several fundamental questions arise:

1. Are the involved probabilistic functions differentiable (in order to apply numerical schemes)?

2. Is the feasible set convex ?

3. Is the feasible set connected ?

4. The tractability of computations in case of formulation in terms of expectation functions and the question of convexity in the lagrangian formulation.

All these questions open a wide area of publications and works. The first question of differentiability involves the choice of the formulation and will be discussed in 7.6.1. More generally, the question of differentiability is treated in [57]. Questions about convexity and connectedness of the feasible set has different answers according to the hypothesis on the stochastic distribution. First, the connectedness of the feasible set is treated by Prékopa in [50] and extended in [30] by Henrion. A growing number of studies explore the question concerning joint probabilities, where convexity and even connectedness of the feasible set can be lost. The present case is rather simple since there is only one constraint. Moreover, the function g is quasi-concave in both arguments [1] since linear, and the normal distribution is log-concave. The formulation is identical to the portfolio optimization given above, and in the chapter 4, [55] presents results concerning convexity of the feasible set under these assumptions. The remaining questions of tractability of the computations and the convexity of the lagrangian formulation are indeed the main directions investigated in this work. These questions are presented in section 7.4. Usually, the question of tractability of computation is linked to the technic of approximation. [55] shows in chapter 4 that in the case of a normal distribution and with a quasi-concave function g, a reformulation gives a convex optimization problem (for instance replace the constraint $G(x) = Pr\{g(x,\chi) \leq c\}$ by $\tilde{G}(x) = ln(p) - ln(Pr\{g(x,\chi) \leq c\}))$. Without modifying the formulation, instead of convex formulation, it is possible to replace concavity assumptions by differentiability requirements. In that case, smoothing technics [57] provide a global frame for numerical computations. A presentation of this approach is given in [10] and [5]. In our approach, we need to preserve direct convexity properties of the original formulation on some subpart of the feasible set in order to use a stochastic algorithm and a sampling method. The fact that convex properties are preserved only on a subpart of the feasible set is due to structural reasons. In [5], the following example is given :

$$\min_{x\in\mathbb{R}} \frac{1}{2}(u-1)^2 \tag{7.10}$$

$$\text{s.t.} \qquad P\{u \leq \chi\} \geq p \tag{7.11}$$

with $p = 0.7$ and a normal distribution χ with $\mu = -2$ and $\sigma = 0.1$; there exist two separate zones of attraction, one associated to the optimal solution $u = -2.05244$, and another one associated with an asymptotic limit $u = 1$.

The reformulation that we propose is not an approximation method and preserves exact computation of the gradient, as far as sampling methods fulfill conditions of convergence. Sampling approximation methods based on Monte Carlo method are commonly used for approximation results [59]. Any sampling can

[1] A quasi-concave function f defined on a convex set X verifies: $\forall(x,x') \in X^2, \forall\lambda \in [0,1], f(\lambda x + (1-\lambda)x') \geq min(f(x), f(x'))$.

be interpreted as an empirical distribution, by replacing the initial constraint $P\{g(x,\chi) \leq c\} \geq p$ by a set of sampling constraints : $P\{g(x,\chi_i) \leq c\} \geq p, i \in [1,\ldots,n]$ with $\chi_1,\ldots,\chi_n$ n independent and identically distributed realizations of χ. Then, according to theoretical analyzis devoted to the tightness of approximation, the number of samples according to the value of p, the rate of convergence, the approximation method are investigated in several works [2],[15],[44]. In our work, this question is viewed from the point of view of the convergence of the stochastic Arrow-Hurwicz algorithm, which is proved in [14]. In 7.4.2, we check all the assumptions to prove that this approach is efficient.

7.3 Formulation with expected values functions and introduction of a lagrangian

The constraint (7.2) can be equivalently reformulated as:

$$\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))\right] \geq p, \tag{7.12}$$

where $\mathbb{H}_{\mathbb{R}^+}$ denotes the indicator function of $\mathbb{R}^+$. Without loss of generality, we assume that $\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - \chi_i)\right] \geq p$ for all $i \in \{1,\ldots,n\}$.

Then, it is possible to formulate the problem and the lagrangian associated with expected values :

$$\mathcal{L}(x,\lambda) \quad = \mathbb{E}\left[\textstyle\sum_{i=1}^n r_i\chi_i x_i\right] - \lambda\left(p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))\right]\right),$$

where λ is the Lagrange multiplier. We call J the objective function of the above maximization problem defined by $J(x) = \mathbb{E}[\sum_{i=1}^n r_i\chi_i x_i]$. We denote $j(x,\chi) = \sum_{i=1}^n r_i\chi_i x_i$. Furthermore, we refer to the function on the left-hand side of the constraint as Θ and to the function inside the expectation of Θ as θ, i.e. $\Theta(x) = \mathbb{E}[\theta(x,\chi)] = \mathbb{E}[\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))]$.
Finally we note: $\mathcal{L}(x,\lambda) = \mathbb{E}[l(x,\lambda,\chi)]$ with $l(x,\lambda,\chi) = \sum_{i=1}^n r_i\chi_i x_i - \lambda\theta(x,\chi)$.
The lagrangian can be written now :

$$\mathcal{L}(x,\lambda) = \mathbb{E}\left[\sum_{i=1}^n r_i\chi_i x_i\right] - \lambda\Theta(x). \tag{7.13}$$

This formulation will be called "expected constraint knapsack problem : "$ECKP$"

Any item that does not satisfy this constraint could be excluded from the beginning. It follows that the optimal solution vector x^* has at least one non-zero component.

Throughout this work, we assume that the weights are normally distributed. Normally distributed items have the nice property that their linear combination is still normally distributed. This property ensures easier numerical computations but is not a foremost condition. To some extent, in the case of normal distributions, it is even possible to compute directly the gradient of the expectation function. Assuming normal distribution is mainly a practical frame in order to produce numerical results and theoretical formulations. Moreover, as mentioned before, normal distribution and a linear constraint are sufficient to have a convex feasible set.

Normally distributed random variables can have negative realizations, and assuming normally distributed weights might thus seem contradictory with the fact that item weights are always strictly positive. However, in most applications, the standard deviation is several times smaller than mean values of the unknown parameters. In this case, the probability of negative weights becomes negligible.

7.3.1 The stochastic gradient type algorithm

We propose to solve the relaxed version of $ECKP$ with a Stochastic Arrow-Hurwicz algorithm (called hereafter SAH-algorithm)) that uses Lagrangian multipliers to deal with the probability constraint.

The SAH-Algorithm can be formulated as follows :

$$(r^k + \lambda^{k-1}(\tau^k)) = \nabla_x l(x^k, \lambda^k, \chi^k)$$

and

$$\theta(x^{k+1}, \chi^k) - p = l'_\lambda(x^{k+1}, \lambda^k, \chi^k)$$

where r^k, λ^k and τ^k are defined in Algorithm 7.3.1:

<u>Stochastic Arrow-Hurwicz Algorithm</u>

1. Choose $x^0 \in X^{ad}_{cont}$ and $\lambda^0 \in [0, \infty)$ as well as two σ-sequences $(\epsilon^k)_{k\in\mathbb{N}^*}$ and $(\rho^k)_{k\in\mathbb{N}^*}$. Set $k = 1$.[2]

2. Given x^{k-1} and λ^{k-1}, draw χ_k following its normal distribution, compute $r^k = \nabla_x j(x^{k-1}, \chi^k)$, $\tau^k = \nabla_x \theta(x^{k-1}, \chi^k)$ and update x and λ as follows:

$$x^k = x^{k-1} + \epsilon^k(r^k + \lambda^{k-1}(\tau^k))$$
$$\lambda^k = \lambda^{k-1} + \rho^k(p - \theta(x^k, \chi^k)).$$

3. If $\lambda^k < 0$ set $\lambda^k = 0$.

[2] a σ-sequence is a sequence of positive numbers, such that $\sum_k \epsilon^k = +\infty$ and $\sum_k (\epsilon^k)^2 < +\infty$.

4. If $k = k_{max}$: STOP. Else: Set $k = k + 1$. Go to step 2.

Note that in the deterministic form of the Arrow-Hurwicz algorithm we would have to use the gradients of the Lagrangian itself. But this function is here an expectation function and its gradient is thus difficult to compute. By drawing independent samples of the random variables at each iteration of the algorithm, the expectation of the gradient is approximated (see [14]), as do Monte Carlo based methods. We discuss in section 7.4 theoretical requirements needed to prove the convergence of the algorithm.
At each iteration of the algorithm we need to calculate the gradient of θ at (x^k, χ^k). However, θ is an indicator function and therefore non-differentiable. In section 7.4, we present a way to bypass this disadvantage by reformulating the constraint function Θ using Integration by Parts.

7.4 Analysis of the convergence

The stochastic gradient algorithm (SAH) used to solve this problem needs to evaluate the gradient of the expected value function, which is an indicator function $\mathbb{H}_{\mathbb{R}^+}(\cdot)$. In [39], this evaluation has been done by using an approximation by convolution, which is a smoothing technic. This method belongs to the class of approximation as mentioned in the introduction. In this work, we study a different approach, a non-biased estimator based on Integration by Parts (called hereafter *IP-method*). The Integration by Parts leads to replace the expected function - which is an indicator function - by a more regular function, that allows a gradient computation. A Heaviside function is piecewise C^∞ and a primitive function is continuous and piecewise derivable, except on zero. So, instead of replacing $\{0,1\}^n$ by $[0,1]^n$ when relaxing $ECKP$, the theoretical analysis will compel us to consider a complementary set of a neighborhood of $0_{[0,1]^n}$. Considering that an empty knapsack is not an optimal solution, it follows that the optimal solution vector of the continuous problem contains at least one component x_κ with $x_\kappa \geq 1/n$. We are thus allowed to replace $[0,1]^n$ by $\{x \in [0,1]^n \mid ||x||_\infty \geq 1/n\} X_{cont}$. Accordingly, we obtain the following feasible set of the relaxed $ECKP$:

$$X_{cont}^{ad} = \{x \in X_{cont} : \Theta(x) \geq p\}.$$

This approach is presented before checking the theorem on convergence of SAH algorithm.

7.4.1 Computation of the gradient of θ

The IP-method involves Integration by Parts to reformulate $\mathbb{E}[\theta(x,\chi)]$ and to obtain an expected value function $\mathbb{E}[\widetilde{\theta}(x,\chi)]$ such that $\mathbb{E}[\widetilde{\theta}(x,\chi)] = \mathbb{E}[\theta(x,\chi)] = \Theta(x)$. $\widetilde{\theta}$

is differentiable and the idea is to use the gradient of $\widetilde{\theta}$ in the SAH-algorithm. [5] present a general method to compute the gradient of an indicator function in expectation using Integration by Parts (see Theorem 5.5 in [**?**]). We state and prove the theorem for the case of $ECKP$. The variables and functions used in this proposition are defined in section 7.3:

Proposition 40 *Let $\mathbb{Y}_{\mathbb{R}^+}(\cdot)$ be a primitive function of $\mathbb{H}_{\mathbb{R}^+}(\cdot)$. Let $x \in X^{ad}_{cont}$ and let $\kappa = \kappa(x) \in \{1, \ldots, n\}$ be defined such that $x_\kappa = ||x||_\infty \geq 1/n$. Then, using Integration by Parts, we get*

$$\Theta(x) = \mathbb{E}\left[\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))M_\kappa(x,\chi)\right]$$

where

$$M_\kappa(x,\chi) = -\frac{(\chi_\kappa - \mu_\kappa)}{\sigma_\kappa^2}\frac{1}{x_\kappa}.$$

It follows that

$$\nabla_x\Theta(x) \;=\; \mathbb{E}\left[-\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))M_\kappa(x,\chi)\chi + \mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))\nabla_x M_\kappa(x,\chi)\right].$$

Proof. Let φ denote the density function of the random vector $\chi = (\chi_1, \ldots, \chi_n)$ and define

$$u'_{\chi_\kappa}(x,\chi) - \mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))x_\kappa \qquad \text{and}$$
$$v(x,\chi) - \frac{\varphi(\chi)}{x_\kappa}.$$

It follows

$$\Theta(x) = \int_{-\infty}^{\infty} \mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))\varphi(\chi)\,\mathrm{d}\chi = \int_{-\infty}^{\infty} u'_{\chi_\kappa}(x,\chi)v(x,\chi)\,\mathrm{d}\chi.$$

Integration by Parts over χ_κ leads to

$$\begin{aligned}\Theta(x) &= \left[u(x,\chi)v(x,\chi)\right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} u(x,\chi)v'_{\chi_\kappa}(x,\chi)\,\mathrm{d}\chi \\ &= -\int_{-\infty}^{\infty} u(x,\chi)v'_{\chi_\kappa}(x,\chi)\,\mathrm{d}\chi = -\int_{-\infty}^{\infty} \mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi))v'_{\chi_\kappa}(x,\chi)\,\mathrm{d}\chi.\end{aligned}$$

In our case the random variables are independently distributed. With φ^i the density function of χ_i we get

$$\begin{aligned}\varphi'_{\chi_\kappa}(\chi) &= \prod_{i\neq\kappa}\varphi^i(\chi_i)\cdot(\varphi^\kappa)'_{\chi_\kappa}(\chi_\kappa) = \prod_{i\neq\kappa}\varphi^i(\chi_i)\cdot\left(-\frac{(\chi_\kappa-\mu_\kappa)}{\sigma_\kappa^2}\varphi^\kappa(\chi_\kappa)\right)\\ &= -\frac{(\chi_\kappa-\mu_\kappa)}{\sigma_\kappa^2}\varphi(\chi).\end{aligned}$$

It follows

$$v'_{\chi_\kappa}(x,\chi) = \frac{\partial}{\partial\chi_\kappa}\left(-\frac{\varphi(\chi)}{x_\kappa}\right) = \frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\varphi(\chi)$$

and therefore

$$\begin{aligned}\Theta(x) &= -\int_{-\infty}^{\infty}\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\varphi(\chi)\,\mathrm{d}\chi\\ &= \mathbb{E}[-\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}].\end{aligned}$$

If $\nabla_x\Theta(x) = \mathbb{E}[\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\right)]$ we get the announced result. Thus, it remains to proof the following result:

Claim 41

$$\nabla_x\Theta(x) = \mathbb{E}[\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\right)]$$

.

This is a classical result under the assumption of uniform boundedness of the gradient function under the integral sign. In our case, we can easily show that this bounding is obtained with the assumption that $x_\kappa \geq 1/n$. First of all, let us observe that we can choose $\mathbb{Y}_{\mathbb{R}^+}(x) = \mathbb{H}_{\mathbb{R}^+}(x)\cdot x = [x]^+$. Therefore, the κ-th component of $\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\right)$ can be reformulated as follows:

$$\begin{aligned}&\left(\nabla_x\left(-\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\right)\right)_\kappa\\ &= \mathbb{H}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\chi_\kappa + [c-g(x,\chi)]^+\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa^2\sigma_\kappa^2}\\ &= \mathbb{H}_{\mathbb{R}^+}(c-g(x,\chi))(\chi_\kappa-\mu_\kappa)\left(\frac{\chi_\kappa}{x_\kappa\sigma_\kappa^2}+\frac{(c-g(x,\chi))}{x_\kappa^2\sigma_\kappa^2}\right).\end{aligned}$$

For $(\chi_\kappa - \mu_\kappa) > 0$ and as $[(c - g(x,\chi)) \geq 0$ is equivalent to $\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi)) = 1]$ it follows

$$0 \leq (\nabla_x \left(-\mathbb{Y}_{\mathbb{R}^+}(c - g(x,\chi)) \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \right))_\kappa \leq (\chi_\kappa - \mu_\kappa) \left(n \cdot \frac{\chi_\kappa}{\sigma_\kappa^2} + n^2 \cdot \frac{(c - g(x,\chi))}{\sigma_\kappa^2} \right)$$

. If $(\chi_\kappa - \mu_\kappa) < 0$, the bounds are reversed.

For the other components $h \neq \kappa$ of the gradient, we just have $\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi)) \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2} \chi_h$ and the same limitations hold for the h-th component. ■

7.4.2 Convergence of the Stochastic Arrow-Hurwicz algorithm

Culioli and Cohen showed in [14] how to obtain the convergence of the SAH-algorithm in the case where the objective function J is convex despite the fact that j is not (for a global survey see [10]). More precisely, they adapt the classical SAH-algorithm to the case where the constraint is given an expectation by considering a subgradient both in dual variables x and λ instead only in λ. We introduce two lemmas to show that the hypothesis needed to apply the SAH algorithm are correctly assumed in the present case, and we discuss some specific particularities. The following lemma proves the assertions on $\tilde{\theta}$ and Θ that must be verified to apply the algorithm.

Assertions on $\tilde{\theta}$ and Θ

Lemma 42 *$\tilde{\theta}(\cdot,\chi)$ is differentiable with a gradient uniformly bounded with respect to χ.*

2. *$\forall \chi \in \mathbb{R}^n$, $\theta(\cdot,\chi)$ is locally Lipschitz continuous.*
3. *There exist $c_1, c_2 > 0$ such that:*
   $\forall \chi \in \mathbb{R}^n$, $\forall x, y \in X_{cont}$, $||\theta(x,\chi) - \theta(y,\chi)|| \leq c_1||x - y|| + c_2$.
4. *$\Theta(.)$ is Lipschitz continuous and concave on a partial set in X_{cont}.*
5. *There exist $\alpha, \beta > 0$ such that:*
   $\forall \chi \in \mathbb{R}^n$, $\forall (x,\tilde{x}) \in X_{cont}^2$ $||\nabla_x j(x,\chi)|| \leq \alpha||x - \widetilde{x}|| + \beta$.
6. *There exist $\gamma, \delta > 0$ such that*
   $\forall x \in X_{cont}$ $\mathbb{E}[\theta(x,\chi) - \Theta(x)]^2 < \gamma||x - \widetilde{x}||^2 + \delta$.

Proof.

1. As presented in subsection 7.6.1, it is possible to replace $\theta(x,\chi)$ by a differentiable function $\widetilde{\theta}(x,\chi)$ such that $\mathbb{E}[\widetilde{\theta}(x,\chi)] = \mathbb{E}[\theta(x,\chi)]$. We get the

following gradient:

$$\nabla_x \widetilde{\theta}(x,\chi) = \mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi))\frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa \sigma_\kappa^2}\chi + \\ [c - g(x,\chi)]^+ \frac{(\chi_\kappa - \mu_\kappa)}{x_\kappa^2 \sigma_\kappa^2}\nu^\kappa$$

2. The fact that $\mathbb{H}_{\mathbb{R}^+}(c - g(x,\chi)) = 1$ only for $c - g(x,\chi) \geq 0$ limits χ to a compact domain. Moreover, adding the fact that $x \in X_{cont}$ where $x_\kappa \geq \frac{1}{n}$ for all $\chi \in \mathbb{R}^n$, $\widetilde{\theta}(\cdot,\chi)$ is locally Lipschitz continuous since $\nabla_x \widetilde{\theta}(x,\chi)$ is bounded. We get that conditions on $\theta(.,\chi)$ are checked.

3. Since $\forall\, \chi \in \mathbb{R}^n$, $\forall\, x, y \in X_{cont}$ $||\theta(x,\chi) - \theta(y,\chi)|| \leq c_2$. Any value of $c_1 > 0$ is correct.

4. Since we assume the item weights to be normally distributed, $\Theta(x) = P\{g(x,\chi) \leq c\}$ is Lipschitz continuous: Let F be the cumulative distribution function of the standard normal distribution. Then we have
$$\Theta(x) = F(\frac{c - \sum_{i=1}^n \mu_i x_i}{\sqrt{\sum_{i=1}^n \sigma_i^2 x_i^2}}).$$
It is easy to see that Θ is continuous on X_{cont}. In addition, we have $0 \leq \Theta(x) \leq 1$ for all $x \in \mathbb{R}^n$ and, as X_{cont} is a compact set, it follows that Θ is Lipschitz continuous on X_{cont}.

5. It is possible to avoid the use of a specific argument based on the normal distribution to establish this assumption. As we replaced θ by $\widetilde{\theta}$, we get:
$$\forall \chi, \forall (x,y) \in X_{cont}\ ||\widetilde{\theta}(x,\chi) - \widetilde{\theta}(y,\chi)|| \leq \tau ||x - y||,$$
where τ bounds $||\nabla_x \widetilde{\theta}||$ given in 1. on X_{cont}. Then we integrate the inequality and we get the bound on Θ.

6. j is linear in each component of x, then there exist $\alpha, \beta > 0$ such that $\forall\, \chi \in \mathbb{R}^n$, $\forall\, (x,\tilde{x}) \in X_{cont}^2$ $||\nabla_x j(x,\chi)|| \leq \alpha ||x - \widetilde{x}|| + \beta$.
Finally we have
$$\theta(x,\chi) \leq 1 \quad \text{and} \quad \Theta(x) = P\{g(x,\chi) \leq c\} \geq 0 \\ \Rightarrow \theta(x,\chi) - \Theta(x) \leq 1 \Rightarrow \mathbb{E}[\theta(x,\chi) - \Theta(x)]^2 \leq 1.$$
It follows that the last requirement is fulfilled for all $\delta > 1$.

■

Concavity of the constraint

The question of concavity of Θ is more complex and depends on the stochastic distribution. There are two different questions to answer: is the constraint function concave on the hypercube, and if not, is the constraint function concave on a partial subset of the hypercube? The answer to the first question is definitively no in a general case [50]. But, we mentioned that very simple chance constrained problems can have several zones of convergence using the lagrangian method. So, indeed, we need to focus on the second question, which depends on initial parameters.

A practical approach combined with a theoretical analysis of the concavity is then possible. We begin to compute the eigenvalues of the Hessian matrix $H(\widetilde{x})$ at a point $\tilde{x}$ close to 0 of the set X_{cont}, and we find them all negative. According to the fact that the determinant of the Hessian matrix $\det(H(x))$ is a continuous function of x, it follows that there exists a connected component $C \subseteq X_{cont}$ of $\widetilde{x}$ such that $\det(H(x)) \neq 0$ for all $x \in C$ and $\widetilde{x} \in C$. We consider the boundary $\overline{C} \backslash C$ of this set and we set $p = \min\{\Theta(x) : x \in \overline{C} \backslash C\}$.
To conclude on the above observations, we formulate the following lemma:

Lemma 43 *Let $\chi_1, ..., \chi_n$ be a fixed set of independently normally distributed random variables and let c be a fixed capacity such that for the vector x with all components near to 0, the Hessian matrix has all eigenvalues strictly negative. Then there exists $p = p(\chi_1, ..., \chi_n, c)$ such that $\Theta(.)$ is concave on the set $\{x \in X_{cont} | \Theta(x) \geq p\}$.*

In other words, we establish that for a given sample set, there exists p such that the corresponding Lagrangian relaxation is concave on the feasible set.

Convexity of J

We suppose that the Lagrangian function admits a saddle point $(\tilde{x}, \tilde{\lambda})$. Generally, strict convexity of J is a sufficient condition to get the stability of the Lagrangian function and the existence of an unique saddle point. In our case, the function J is only linear. In deterministic optimization, the problem of lack of strict concavity of the objective function is often solved by applying an augmented Lagrangian method. This approach consists in subtracting a strictly convex function of the constraint function (generally the square of the euclidean norm) from the simple Lagrangian. In our case, we would obtain the following augmented Lagrangian:

$$\begin{aligned}
\mathcal{L}_{aug}(x, \lambda) &= \mathbb{E}[j(x, \chi)] - \frac{1}{2\gamma} \left(\| \left[\lambda + \gamma(p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right])\right]^+ \|^2 - \| \lambda \|^2 \right) \\
&= \mathbb{E}[j(x, \chi)] - \frac{1}{2\gamma} \left(\left(\left[\lambda + \gamma(p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c - g(x, \chi))\right])\right]^+\right)^2 - \lambda^2 \right) \quad (7.14) \\
&= \mathcal{L}(x, \lambda) + \frac{\gamma}{2} \mathbb{E}(\theta(x))^2. \quad (7.15)
\end{aligned}$$

with a constant $\gamma > 0$. Unfortunately, the term $\frac{\gamma}{2}\mathbb{E}(\theta(x))^2$ is non-linear, and computing its gradient is out of the scope of this work.

7.5 Technical Implementation of the SAH algorithm

Some improvements in our implementations are worth being noted:

- During our first numerical tests we remarked that the SAH-algorithm involving the IP-method was inefficient and even did not converge in some cases. Then we analyzed and modified the implementation of the IP-method. The first enhancement deals with the formulation of $ECKP$. The expectation constraint states

$$\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c-g(x,\chi))\right] \geq p \Leftrightarrow p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c-g(x,\chi))\right] \leq 0.$$

We thus get the Lagrangian

$$\mathcal{L}(x,\lambda) = \mathbb{E}\left[\sum_{i=1}^{n} r_i\chi_i x_i\right] - \lambda\left(p - \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(c-g(x,\chi))\right]\right).$$

Using the IP-method, we rewrite this Lagrangian as follows:

$$\mathcal{L}(x,\lambda) = \mathbb{E}[\sum_{i=1}^{n} r_i\chi_i x_i] - \lambda\left(p + \mathbb{E}[\mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}]\right). \quad (7.16)$$

Let us denote $\widetilde{l}$ the function inside the expectation of the Lagrangian (7.16), i.e.

$$\widetilde{l}(x,\lambda,\chi) = \sum_{i=1}^{n} r_i\chi_i x_i - \lambda\left(p + \mathbb{Y}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\right).$$

It follows

$$\left(\nabla_x\widetilde{l}(x,\lambda,\chi)\right)_h = r_h\chi_h + \lambda\left(\mathbb{H}_{\mathbb{R}^+}(c-g(x,\chi))\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\left(\chi_h + \frac{(c-g(x,\chi))}{x_\kappa}\nu_h^\kappa\right)\right).$$

Observe that the term that multiplies λ is zero whenever the capacity constraint *is not* satisfied. In these cases *all* the components of the current x^k are incremented (as all the components of $(r_1\chi_1,\ldots,r_n\chi_n)^T$ are positive) although at least one component should be decremented in order to better fit the capacity.
The constraint in expectation can be equivalently reformulated as

$$\mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(g(x,\chi)-c)\right] \leq 1-p \Leftrightarrow \mathbb{E}\left[\mathbb{H}_{\mathbb{R}^+}(g(x,\chi)-c)\right] - (1-p) \leq 0.$$

In this case the hth component of the gradient of $\widetilde{l}$ is

$$\left(\nabla_x\widetilde{l}(x,\lambda,\chi)\right)_h = r_h\chi_h - \lambda\left(\mathbb{H}_{\mathbb{R}^+}(g(x,\chi)-c)\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\left(\chi_h - \frac{(g(x,\chi)-c)}{x_\kappa}\nu_h^\kappa\right)\right).$$

Here the term that multiplies λ is zero whenever the capacity constraint *is* satisfied. In this case the components of x^k are incremented by the components of the positive vector $(r_1\chi_1^k, \ldots, r_n\chi_n^k)^T$ (multiplied by the corresponding factor σ^k). When the capacity constraint is not satisfied the term with coefficient λ is subtracted from this vector in order to correct x^k. This term is positive whenever $(\chi_\kappa^k - \mu_\kappa) > 0$ and the Lagrange multiplier λ is thus playing its assigned role of a penalty factor in the case where the constraint is violated.

- A second improvement can be obtained by a specific choice of the component to be integrated by parts: The hth component of the gradient of the Lagrangian function obtained by the IP-method after reformulation:

$$\left(\nabla_x \widetilde{l}(x,\lambda,\chi)\right)_h = r_h\chi_h - \lambda\left(\mathbb{H}_{\mathbb{R}^+}(g(x,\chi)-c)\frac{(\chi_\kappa-\mu_\kappa)}{x_\kappa\sigma_\kappa^2}\left(\chi_h - \frac{(g(x,\chi)-c)}{x_\kappa}\nu_h^\kappa\right)\right).$$

In case of an overload, we expect this gradient to be negative for some indexes h in order to decrease the total weight of the knapsack. However, for $h \neq \kappa$ and $\mathbb{H}_{\mathbb{R}^+}(g(x,\chi)-c) = 1$, the term that multiplies λ is positive if and only if $(\chi_\kappa - \mu_\kappa) > 0$, which is not always the case. When this does not occur, the gradient is strictly positive and *all* components of x with index different from κ are incremented despite the overload. Due to this observation we propose the following improved choice of the index κ: Instead of just choosing κ in the kth iteration such that $x_\kappa^k = ||x^k||_\infty$ (see Proposition 40), we choose κ as follows:

$$\kappa = \arg\max_{i=1,\ldots,n}\{x_i^k | x_i^k \geq 1/n \text{ and } (\chi_i^k - \mu_i) > 0\}$$

However, if $\{x_i^k | x_i^k \geq 1/n \text{ and } (\chi_i^k - \mu_i) > 0\} = \emptyset$, we choose κ as before, i.e. such that $x_\kappa^k = ||x^k||_\infty \geq 1/n$.
With this modification, we significantly improved the convergence of the *SAH*-algorithm involving the IP-method and could reduce the maximum number of iterations to 1000 with effective results.

7.6 Numerical results

7.6.1 Convergence of the Stochastic Arrow-Hurwicz Algorithm

In the case of gradient class algorithm, the speed of variation of the objective value cannot be efficiently used to establish a stopping rule of the algorithm. A classical approach in this case leads to set a maximum number of iterations. We fixed our limitation to 1000 iterations.(see Table 7.1).

In order to evaluate the efficiency of our computations, we compare the Integration By Parts with a simple method of computation that consists in approximating the hth component of the gradient of θ by the corresponding differences ratio:

$$\frac{\theta(x+\delta\nu^h,\chi)-\theta(x-\delta\nu^h,\chi)}{2\delta},$$

where $\delta > 0$ and $\nu^h \in \{0,1\}^n$ such that $\nu^h_h = 1$ and $\nu^h_i = 0$ for $i \neq h$. This leads to the following approximation of the hth component of the gradient of θ:

$$(\nabla_x(\theta_\delta)(x,\chi))_h = \frac{\mathbb{H}_{\mathbb{R}^+}(c-g(x+\delta\nu^h,\chi)) - \mathbb{H}_{\mathbb{R}^+}(c-g(x-\delta\nu^h,\chi))}{2\delta}.$$

All numerical tests have been carried out on an Intel PC with $2GB$ RAM. The SAH-algorithm algorithms as well as the branch-and-bound framework have been implemented in C language. The random numbers needed for the computations of the stochastic gradient algorithm where generated in advance using the gsl C-library and stored in a file. We tested the SAH-algorithm on the instance used in [11] called hereafter C./B., as well as on 50 randomly generated instances for each dimension. The data given in the tables 7.1 and 7.2 are average values over these instances. As noticed at the end of section 7.3, the choice of normally distributed weights implies that theoretically weights can take negative values. The test instances were generated in such a way that the variance/mean ratio is below 1/4 (see [11] or [39] for details). This implies a very low probability for the realization of negative weights: Although a high number of scenarios were generated (either 500 or 1000 for each run of the SAH-algorithm), we have not encountered any negative weight realization.
In Table 7.1, the numerical results of the SAH-algorithm involving the FD- or IP-method are compared with those using a *Second Order Cone Programming*-solver ($SOCP$): As mentioned, $ECKP$ can be equivalently reformulated as a chance constrained knapsack problem that, in turn, can be reformulated as a deterministic equivalent $SOCP$-problem (for details see [7] or [39]). The deterministic equivalent formulation is closely connected with the Prékopa's result that for $p > \frac{1}{2}$, the feasible set is convex. Remind that we call F the cumulative distribution function of the standard normal distribution. Then we have

$$\Theta(x) = F\left(\frac{c-\sum_{i=1}^n \mu_i x_i}{\sqrt{\sum_{i=1}^n \sigma_i^2 x_i^2}}\right) \geq p \Leftrightarrow \frac{c-\sum_{i=1}^n \mu_i x_i}{F^{-1}(p)} \geq \sqrt{\sum_{i=1}^n \sigma_i^2 x_i^2}$$

This formulation can be seen as the condition that a vector x is closer to the origin than to an hyperplane for some euclidean distance. This type of set is the interior of a paraboloid.
First of all, we remark that the optimal solution values of the SAH-algorithm involving the FD-method are comparable with those produced by the IP-method.

Some fluctuations occur and can be interpreted in terms of choice of implementation, like the choice of the two σ-sequences for the SAH-algorithm. Choosing the right parametrization for these sequences has an important influence on the convergence of the algorithm and the best found solution.
In terms of running time, both methods outperform the $SOCP$-algorithm for small and medium size instances. For higher dimensional problems, this is still true for the FD-method. However, the SAH-algorithm involving the IP-method needs approximately twice the time as when using the FD-method. This is of course due to the total number of iterations that we fixed at 500 when using the FD-method, while we fixed 1000 iterations with the IP-method in order to obtain equally good solutions. For higher dimensional instances, the Mosek interior point method needs therefore less CPU-time than the IP-method.

7.7 Solving the (combinatorial) ECKP - Numerical Results

The combinatorial problem has been solved using a branch-and-bound algorithm as described in [11] and [39]. The algorithm has been tested on the instances previously used for the tests of the SAH-algorithm.
We stored the random numbers needed for the executions of the SAH in a batch file. As the total number of executions during the branch-and-bound algorithm is unknown and the number of random numbers needed for all those executions is generally very high, we only stored random numbers for a limited number of executions. Before starting an execution of the SAH we then chose randomly one of the instances of random numbers. Remark that, as the executions of the SAH are independent, one stored instance of random numbers would theoretically be sufficient.
The results given in Table 7.2 indicate that the branch-and-bound algorithm that uses $SOCP$-program to obtain upper bounds, considers more nodes than when using an SAH-algorithm. This is not due to a better choice of the upper bounds in the latter method, as in both algorithms the upper bounds are supposed to be the same (i.e. the optima of the corresponding relaxed problems). However, as the best solution found by the approximate SAH-algorithm might be slightly smaller than the solution value of the relaxed problem, more branches are pruned than with the primal-dual $SOCP$-algorithm. Note that this could theoretically also cause the pruning of a subtree that contains the optimal solution.
Similarly, the SAH-algorithm involving the IP-method considers an average number of nodes greater than when using the FD-method. This implies that the solutions of the relaxed subproblems produced by the FD-method are not as good as those obtained when using the IP-method. As both methods perform equally on the relaxed overall problems (see subsection 7.6), we conclude that the FD-method

	Arrow-Hurwicz & FD-method			Arrow-Hurwicz & IP-method			SOCP	
n	**Optimum**	**CPU-time (msec)**	**Gap**	**Optimum**	**CPU-time (msec)**	**Gap**	**Optimum**	**CPU-time (msec)**
C./B.	4695.525	3	0.02 %	4667.790	30	0.75%	4696.413	4
15	4954.314	5	0.01 %	4910.434	30	0.89%	4954.704	4
20	6712.219	8	0.03 %	6653.941	40	0.89%	6713.987	6
30	10308.293	14	0.02 %	10174.878	53	1.31%	10310.45	18
50	16992.444	28	0.01 %	16743.260	92	1.47%	16993.514	65
75	25565.525	52	0.01 %	25155.023	138	1.63%	25569.379	213
100	33902.119	85	0 %	33208.278	181	2.05%	33903.672	503
150	50677.120	174	0 %	49609.044	271	2.10%	50678.312	1802
250	85186.119	438	0 %	83785.657	431	1.64 %	**	**
500	170226.568	1638		167342.988	869		**	**
1000				334984.436	1720		**	**
5000				1676670.246	8640		**	**
20000				6731686.102	36031		**	**

** exceeding of the available memory space

Figure 7.1: Numerical results for the continuous *ECKP*

is less robust: Instead of choosing particular σ-sequences for each instance or even each subproblem that has to be solved during the branch-and-bound algorithm, we fixed one parametrization for each dimension. However, the subproblems solved during the branch-and-bound algorithm are mostly lower dimensional problems. And the SAH-algorithm using the fixed σ-sequences is less performant on these subproblems. This seems to be especially the case when using the FD-method.
If we only allow an average computing time of $1h$, the $SOCP$-algorithm can only be used up to dimension 50. On the contrary, when using the FD-method and allowing 500 iterations in the SAH-algorithm, we are able to solve problems up to dimension 250 within an average CPU-time of $1h$.

7.8 Conclusion

In this part, we have studied and analyzed a method for computing the gradient of the Stochastic Knapsack Problem, where the expectation constraint is considered. The constraint expressed here is scalar. Additional work is necessary to study the case of joint probabilistic constraints. As mentioned in the introduction, the joint probabilistic constraints formulation is given by :

$$\max f(x) \tag{7.17}$$

$$\text{s.t.} \quad P\{g_j(x,\chi) \leq 0, j \in \mathbb{J}\} \geq p \tag{7.18}$$

$$x \in \mathbb{X} \tag{7.19}$$

where $\mathbb{X} \subset \mathbb{R}^n$ is non-empty, $f : \mathbb{R}^n \to \mathbb{R}$, $g_j : \mathbb{R}^n \times \mathbb{R}^s \to \mathbb{R}$, $\mathbb{J}$ is a set of more than two indexes, and χ is a s-dimensional random vector. In the case of the stochastic Knapsack, this means the introduction of a multi-purpose knapsack for instance. This type of formulation will be developed in future works.

7.9 Global Conclusion and future works

Our work has been developed towards two different directions :

- A Two-Stage formulation of the stochastic Maximal Weight Forest Problem.
- A Chance Constrained formulation of the stochastic Knapsack Problem.

In the case of the Maximal Weight Forest Problem, we proved that for some specific cases, the algebraical properties of the Dual formulation are preserved and that the system is Totally Dual Integral. In a general case, we found and analyzed necessary and sufficient conditions to preserve integrity. We proposed an algorithm of approximation based on these conditions. Finally, we proposed a compact model formulation for the exponential system of constraints associated to a covering forest and we presented numerical results. Moreover, we introduced new functions to

	Arrow-Hurwicz & FD-method				Arrow-Hurwicz & IP-method				SOCP		
n	Optimum	Number of nodes	CPU-time (sec) B-and-B	Gap	Optimum	Number of nodes	CPU-time (sec) B-and-B	Gap	Optimum	Number of nodes	CPU-time (sec) B-and-B
C./B.	4595	122	0.09	0 %	4595	122	0.21	0 %	4595	122	0.406
15	4840	31	0.02	0 %	4840	34	0.06	0 %	4840	34	0.082
20	6634	69	0.07	0 %	6634	63	0.13	0 %	6634	66	0.236
30	10272	271	0.35	0 %	10272	436	1.22	0 %	10272	350	1.801
50	16975	3341	6.43	0 %	16975	7051	31.34	0 %	16975	7406	70.914
75	25548	6187	18.43	0 %	25548	23911	161.47	0 %	25548	62175	1535.520
100	33895	12093	45.87	-	33894	98479	1049.18	-	*	*	*
150	50672	37076	244.11	-	*	*	*	*	*	*	*
250	85189	65890	623.53	-	*	*	*	*	*	*	*
500	*	*	*	*	*	*	*	*	*	*	*

Figure 7.2: Numerical results for the (combinatorial) $ECKP$

handle the difficulty that in case of minimal fluctuations in the cost functions, the chosen forest should drastically change. We considered that the status of an edge chosen or not in the greedy solution is an implicit function depending of its cost, but equally of the cost of the other edges, and of the status of these other edges. We reintroduced some continuous dependence into the greedy solution when the cost function is slightly modified. We believe that this approach can be extended to several combinatorial problems with discrete stochastic distributions. We will investigate some improvements in the approximation algorithm and try to perform it on larger instances.
In the case of the Chance Constrained Knapsack (second part), we analyzed the efficiency of a gradient based method by reformulating an expected constraint with smooth functions considering Integration by Parts on the constraint on expectation. The method proposed outperforms geometrical reformulation based on the properties of normal distributions (SOCP). Many questions arise concerning the efficiency of the numerical stability of these methods, and we should explore these questions. There are several developments that can be foreseen after this work : the same stochastic Knapsack problem with multiple constraints is a joint probability problem. It is possible to attempt to apply similar techniques as Integration by Parts (for instance geometrical transformation) to the expected formulation, in order to compute separately several constraints. Another direction of work is the use of semidefinite programming methods for approximating the NP hard combinatorial stochastic problems that we are trying to investigate with the Shortest Path problem.

Appendix A

Porta

```
 DIM = 10

LOWER_BOUNDS
0 0 0 0 0 0 0 0 0 0
UPPER_BOUNDS
1 1 1 1 1 1 1 1 1 1

INEQUALITIES_SECTION
(  2) x1 + x2 + x4 + x5                  <= 3
(  3) x3 + x6 + x7                   <= 2
(  4) x2 + x3 + x8                     <= 2
(  5) x1 + x3 + x9 + x10                     <= 3


( 27)  x1                           <= 1
( 28)  x2                          <= 1
( 29)  x3                          <= 1
( 30)  x4                          <= 1
( 31)  x5                          <= 1
( 32)  x6                          <= 1
( 33)  x7                          <= 1
( 34)  x8                          <= 1
( 35)  x9                          <= 1
( 36)  x10                          <= 1

( 42)  x1                           >= 0
( 43)  x2                          >= 0
```

```
( 44)  x3                          >= 0
( 45)  x4                          >= 0
( 46)  x5                          >= 0
( 47)  x6                          >= 0
( 48)  x7                          >= 0
( 49)  x8                          >= 0
( 50)  x9                          >= 0
( 51)  x10                         >= 0

END
```

Appendix B

maple

Definition of a procedure f computing the optimal value

```
> with(simplex):
> f:=proc(c1,c2,c3,c4,c5,c6,c7,c8,c9,c10)
local l,s,i,a,x1,x2,x3,x4,x5,x6,x7,x8,x9,X10;
l:=[c1,c2,c3,c4,c5,c6,c7,c8,c9,c10];
assign(maximize(c1*x1+c2*x2+c3*x3+c4*x4+c5*x5+c6*x6+c7*x7+c8*x8+
 c9*x9+c10*x10,
{x1<= 1,x2 <=1,x3<=1,x4<=1,x5<=1,x6<=1,x7<=1,x8<=1,x9 <=1,x10 <=1,x1+x2+x4+x5<=3,
x3+x6+x7<=2,x2+x3+x8<=2,x1+x3+x9+x10<=3},NONNEGATIVE));
s:=c1*x1+c2*x2+c3*x3+c4*x4+c5*x5+c6*x6+c7*x7+c8*x8+c9*x9+c10*x10;
end:
```

Definition of a procedure g computing the optimal vector

```
> g:=proc(c1,c2,c3,c4,c5,c6,c7,c8,c9,c10)
> unassign('x1','x2','x3','x4','x5','x6','x7','x8','x9','x10');
> maximize(c1*x1+c2*x2+c3*x3+c4*x4+c5*x5+c6*x6+c7*x7+c8*x8+c9*x9+c10*x10,
> {x1<= 1,x2 <=1,x3<=1,x4<=1,x5<=1,x6<=1,x7<=1,x8<=1,x9 <=1,x10 <=1,
> x1+x2+x4+x5<=3,x3+x6+x7<=2,x2+x3+x8<=2,x1+x3+x9+x10<=3},NONNEGATIVE);
> end:
```

Optimal value and optimal vector with a cost vector (5,5,5,10,10,10,4,10,10,10)

```
> f(5,5,5,10,10,10,4,10,10,10);
```

$$\frac{139}{2}$$

```
> g(5,5,5,10,10,10,4,10,10,10);
```

$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$

Decreasing the value of c3 to 0, identify the chosen edges.

```
> f(5,5,0,10,10,10,4,10,10,10);
```

$$69$$

```
> g(5,5,0,10,10,10,4,10,10,10);
```

$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

Leveling the value of c3 to 3.9999, identify the chosen edges.

```
> f(5,5,3.99,10,10,10,4,10,10,10);
```

$$69.0$$

```
> g(5,5,3.99,10,10,10,4,10,10,10);
```

$$\{x1 = 1, x10 = 1, x2 = 0, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

Variation of the cost of the third edge from 0 to 7

```
> for i from 0 to 7
> do
> print('c3'=i);
> print('Z[Lp]'=f(5,5,i,10,10,10,4,10,10,10));
> g(5,5,i,10,10,10,4,10,10,10)
> od;
```

$$c3 = 0$$
$$Z_{Lp} = 69$$
$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$
$$c3 = 1$$
$$Z_{Lp} = 69$$
$$\{x1 = 1, x10 = 1, x2 = 0, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$
$$c3 = 2$$
$$Z_{Lp} = 69$$
$$\{x1 = 1, x10 = 1, x2 = 0, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$
$$c3 = 3$$
$$Z_{Lp} = 69$$
$$\{x1 = 1, x10 = 1, x2 = 0, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$
$$c3 = 4$$
$$Z_{Lp} = 69$$
$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$
$$c3 = 5$$
$$Z_{Lp} = \frac{139}{2}$$
$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$
$$c3 = 6$$
$$Z_{Lp} = 70$$

$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$

$$c3 = 7$$

$$Z_{Lp} = \frac{141}{2}$$

$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$

Same work with another first stage edge (x_1)
Variation of the cost of the first edge from 0 to 7

```
> for i from 0 to 7
> do
> print('c1'=i);
> print('Z[Lp]'=f(i,5,5,10,10,10,4,10,10,10));
> g(i,5,5,10,10,10,4,10,10,10)
> od;
```

$$c1 = 0$$

$$Z_{Lp} = 69$$

$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

$$c1 = 1$$

$$Z_{Lp} = 69$$

$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

$$c1 = 2$$

$$Z_{Lp} = 69$$

$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

$$c1 = 3$$

$$Z_{Lp} = 69$$

$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

$$c1 = 4$$

$$Z_{Lp} = 69$$

$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$

$$c1 = 5$$

$$Z_{Lp} = \frac{139}{2}$$

$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$

$$c1 = 6$$

$$Z_{Lp} = 70$$

$$\{x1 = 1/2, x10 = 1, x2 = 1/2, x3 = 1/2, x4 = 1, x5 = 1, x6 = 1, x7 = 1/2, x8 = 1, x9 = 1\}$$

$$c1 = 7$$

$$Z_{Lp} = 71$$

$$\{x1 = 1, x10 = 1, x2 = 0, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

```
> f(3.999,5,5,10,10,10,4,10,10,10);
> g(3.999,5,5,10,10,10,4,10,10,10);
```

$$69.0$$

$$\{x1 = 0, x10 = 1, x2 = 1, x3 = 0, x4 = 1, x5 = 1, x6 = 1, x7 = 1, x8 = 1, x9 = 1\}$$

Bibliography

[1] Semra Agrali and Joseph Geunes. A single-resource allocation problem with Poisson resource requirements. *Optimization Letters*, 3:559–571, 2009.

[2] Shabbir Ahmed and Alexander Shapiro. The Sample Average Approximation Method for Stochastic Programs with Integer Recourse. 2002.

[3] Shabbir Ahmed, Mohit Tawarmalani, and Nikolaos V. Sahinidis. A finite branch-and-bound algorithm for two-stage stochastic integer programs. *Mathematical Programming*, 100:355–377, 2004.

[4] Grégoire Allaire. *Analyse numérique et optimisation.* Editions de l'école Polytechnique, 2009.

[5] Laetitia Andrieu, Guy Cohen, and Felisa Vázquez-Abad. Stochastic programming with probability constraints. http://fr.arxiv.org/abs/0708.0281 (Accessed 24 October 2008), 2007.

[6] Moshe Babaioff, Nicole Immorlica, David Kempe, Robert Kleinberg, M. Charikar, K. Jansen, O. Reingold, and J. Rolim. *A Knapsack Secretary Problem with Applications.* 2007.

[7] Stephen Boyd, Herve Lebret, Miguel Soma Lobo, and Lieven Vandenberghe. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[8] Claus C. Carøe and Jørgen Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83:451–464, 1998.

[9] Claus C. Carøe and Jørgen Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming*, 83:451–464, 1998.

[10] Carpentier. Méthodes numériques en optimisation stochastique. Ecole Nationale Supérieure des Techniques Avancées. http://www.ensta.fr/ pcarpent/MNOS/ (Accessed march 2013), 2010.

[11] Amy Cohn and Cynthia Barnhart. The stochastic knapsack problem with random weights: A heuristic approach to robust transportation planning. 1998.

[12] W. Cook. operations that preserve total dual integrality. *Elsevier Science Publishers B.V*, 2:31–35, 1983.

[13] Gérard Cornuéjols. Review of combinatorial optimization: packing and covering. *Sigact News*, 39:16–18, 2008.

[14] J. C. Culioli and G. Cohen. Optimisation stochastique sous contraintes en espérance. *Comptes rendus de l'Académie des sciences, Paris, Série I*, 320(6):753 ? 758, 1995.

[15] L. Dai, C. H. Chen, and J. R. Birge. Convergence Properties of Two-Stage Stochastic Programming. *Journal of Optimization Theory and Applications*, 106:489–509, 2000.

[16] G B Dantzig and D R Fulkerson. On the max-flow min-cut theorem of networks. *Annals of mathematical studies 38*, pages 215–221, 1956.

[17] K. Dhamdhere, R. Ravi, and M. Singh. On two-stage stochastic minimum spanning trees. *IPCO 2005, LNCS 3509*, pages 321–334, 2005.

[18] B. Escoffier, L.GourvÃÍs, J Monnot, and O Spanjaard. Two-stage stochastic matching and spanning tree problems : polynomial instances and approximation. *preprint Elsevier*, 2009.

[19] A. Flaxman, A. Frieze, and Krivelevich. On the random 2-stage minimum spanning tree. *journal of the Association for Computing Machinery*, preprint 1995.

[20] Fournier. *Introduction à la notion de matroide - géométrie combinatoire*. Publications mathematiques d'Orsay, 1979.

[21] A. Franck. A weighted matroid intersection. *Journal of Algorithms*, 2:328–336, 1981.

[22] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15:835–855, 1965.

[23] M. Garey and D. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness. 1989.

[24] A. Ghouila-houri. Caractérisation des matrices totalement unimodulaires. 1962.

[25] F. R. Giles and W. R. Pulleyblank. Total dual integrality and integer polyhedra. *Linear Algebra and Its Applications*, 25:191–196, 1979.

[26] Ashish Goel and Piotr Indyk. Stochastic Load Balancing and Related Problems. In *IEEE Symposium on Foundations of Computer Science*, pages 579–586, 1999.

[27] A. Gupta, M. Pal, R. Ravi, and A. Sinha. Boosted sampling: approximation algorithms for stochastic optimization. *Proc. of the 36th Annual ACM Symposium on Theory of Computong (STOC'04)*, pages 417–426, 2004.

[28] Willem K. Klein Haneveld, Leen Stougie, and Maarten H. van der Vlerk. An algorithm for the construction of convex hulls in simple integer recourse programming. *Annals of Operations Research*, 64:67–81, 1996.

[29] Willem K. Klein Haneveld and Maarten H. van der Vlerk. Stochastic integer programming:General models and algorithms. *Annals of Operations Research*, 85:39–57, 1999.

[30] R. Henrion. On the Connectedness of Probabilistic Constraint Sets. *Journal of Optimization Theory and Applications*, 112:657–663, 2002.

[31] D. Karger, P.Klein, and R. Tarjan. A randomized linear-time algorithm to find minimum spanning trees. *journal of the Association for Computing Machinery*, 42:321–328, 1995.

[32] V. King. A simpler minimum spanning tree verification algorithm. *Proceedings of the Workshop on Algorithm and Data Structures*, 1995.

[33] Anton J. Kleywegt and Jason D. Papastavrou. The Dynamic and Stochastic Knapsack Problem with Random Sized Items. *Operations Research*, 49:26–41, 2001.

[34] Olivier Klopfenstein and Dritan Nace. A robust approach to the chance-constrained knapsack problem. *Operations Research Letters*, 36:628–632, 2008.

[35] N. Kong and J. Schaefer. A factor 1/2 approximation algorithm for two-stage stochastic matching problems. *European Journal of Operational Research 172*, 3:740–746, 2006.

[36] Nan Kong, Andrew J. Schaefer, and Shabbir Ahmed. Totally Unimodular Stochastic Programs. *Mathematical Programming*, 2012.

[37] Nan Kong, Andrew J. Schaefer, and Brady Hunsaker. Two-stage integer programs with stochastic right-hand sides: a superadditive dual approach. *Mathematical Programming*, 108:275–296, 2006.

[38] Stefanie Kosuch, Marc Letournel, and Abdel Lisser. stochastic knapsack problem . *to be published*, 2013.

[39] Stefanie Kosuch and Abdel Lisser. Upper bounds for the 0-1 stochastic knapsack problem and a BB algorithm. *Annals of Operations Research*, 176:77–93, 2010.

[40] J.B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceeding of the American Mathematical Society*, 7:48–50, 1956.

[41] Gilbert Laporte, François V. Louveaux, and Luc Van Hamme. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50:415–423, 2002.

[42] Grace Y. Lin, Yingdong Lu, and David D. Yao. The Stochastic Knapsack Revisited: SwitchOver Policies and Dynamic Pricing. *Operations Research*, 56:945–957, 2008.

[43] P. Adasme R. Andrade M. Letournel A. Lisser. A polynomial formulation for the stochastic maximum weight forest problem. *to be published*, 2013.

[44] James Luedtke and Shabbir Ahmed. A Sample Approximation Approach for Optimization with Probabilistic Constraints. *Siam Journal on Optimization*, 19:674–699, 2008.

[45] R Martin. Using separation algorithms to generate mixed integer model reformulations. *Operations Research Letters*, 10:119–128, 1991.

[46] R. K. Martin. Using separation algorithms to generate mixed integer model reformulations. *Operations Research Letters 10(3) , 119–128.*, 1991.

[47] Yasemin Merzifonluoglu, Joseph Geunes, and H. Edwin Romeijn. The static stochastic knapsack problem with normally distributed item sizes. *Mathematical Programming*, pages 459–489, 2012.

[48] George Nemhauser and Laurence Wolsey. *Integer And Combinatorial Optimization.* Wiley-Interscience series in Discrete Mathematics and Optimization, 1999.

[49] James G. Oxley. *Matroid theory.* 1992.

[50] A Prékopa. *Stochastic Programming.* Kluwer Dordrecht Holland, 1995.

[51] R. Ravi and A. Sinha. Hedging uncertainty: Approximation algorithms for stochastic optimization problems. *Math. Program.*, 108:97–114, 2006.

[52] W Rudin. *Real and complex analysis.* 1966.

[53] Rudiger Schultz, Leen Stougie, and Vlerk van der M. H. Two-stage stochastic integer programming: a survey. *Statistica Neerlandica*, 50:404–416, 1996.

[54] Ahmed Shabbir. *Tutorials on stochastic programming and applications.* University of Arizona, 1996.

[55] Alexander Shapiro, Darinka Dentcheva, and Andrej Ruszczyinski. *Lectures on Stochastic Programming.* MPS-SIAM Series on Optimization, 2009.

[56] D. Shmoys and C. Swamy. Stochastic optimization is (almost) as easy as deterministic optimization. *Proceedings of 45th Annual IEEE Symposium on Foundations of Computer Science (FOC'S 04)*, 2004.

[57] Stanislav Uryasev. *Introduction to the theory of probabilistic functions and percentiles (value-at-risk).* Kluwer Academic Publishers, 2000.

[58] V. Vazirany. *Approximation Algorithms.* Springer, 2003.

[59] W. Wang. Sample average approximation of risk-averse stochastic programs. phd thesis. Georgia Institute of Technology, Atlanta, GA,, 2007.