50376
1979
105

N° d'ordre : 464

THÈSE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNIQUES DE LILLE

pour obtenir le grade de

DOCTEUR ES SCIENCES MATHÉMATIQUES

par

Didier FAYARD

CONTRIBUTION A LA RESOLUTION DES PROGRAMMES MATHEMATIQUES EN NOMBRES ENTIERS

Soutenue le 28 septembre 1979, devant la Commission d'Examen

MM.	P.	POUZET	Président
	C.	BREZINSKI	Examinateur
	J.C.	FIOROT	Examinateur
	P.	HANSEN	Examinateur
	A.	LAPLACE	Examinateur
	P.	HUARD	Rapporteur

Ce mémoire est commun[illegible]RD et G. PLATEAU

SCD LILLE 1
D 030 194869 1

d'ordre : 464

THÈSE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNIQUES DE LILLE

pour obtenir le grade de

DOCTEUR ES SCIENCES MATHÉMATIQUES

par

Didier FAYARD

CONTRIBUTION A LA RESOLUTION DES PROGRAMMES MATHEMATIQUES EN NOMBRES ENTIERS

Soutenue le 28 septembre 1979, devant la Commission d'Examen

MM.	P.	POUZET	Président
	C.	BREZINSKI	Examinateur
	J.C.	FIOROT	Examinateur
	P.	HANSEN	Examinateur
	A.	LAPLACE	Examinateur
	P.	HUARD	Rapporteur

Ce mémoire est commun aux thèses de D. FAYARD et G. PLATEAU

IIVERSITE DES SCIENCES
TECHNIQUES DE LILLE

01.02.1976

DOYENS HONORAIRES de l'Ancienne Faculté des Sciences

. R. DEFRETIN, H. LEFEBVRE, M. PARREAU.

PROFESSEURS HONORAIRES des Anciennes Facultés de Droit et Sciences Economiques, des Sciences et des Lettres

ARNOULT, Mme BEAUJEU, MM. BROCHARD, CHAPPELON, CHAUDRON, CORDONNIER, CORSIN, DEHEUVELS, HORS, DION, FAUVEL, FLEURY, P. GERMAIN, HEIM DE BALSAC, HOCQUETTE, KAMPE DE FERIET, UGANOFF, LAMOTTE, LASSERRE, LELONG, Mme LELONG, MM. LHOMME, LIEBAERT, MARTINOT-LAGARDE, ZET, MICHEL, NORMANT, PEREZ, ROIG, ROSEAU, ROUBINE, ROUELLE, SAVART, WATERLOT, WIEMAN, MANSKI.

PRESIDENTS HONORAIRES DE L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE

I. R. DEFRETIN, M. PARREAU.

PRESIDENT DE L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE

J. LOMBARD.

PROFESSEURS TITULAIRES

BACCHUS Pierre	Astronomie
BEAUFILS Jean-Pierre	Chimie Physique
BECART Maurice	Physique Atomique et Moléculaire
BILLARD Jean	Physique du Solide
BIAYS Pierre	Géographie
BONNEMAN Pierre	Chimie Appliquée
BONNOT Ernest	Biologie Végétale
BONTE Antoine	Géologie Appliquée
BOUGHON Pierre	Algèbre
BOURIQUET Robert	Biologie Végétale
CELET Paul	Géologie Générale
CONSTANT Eugène	Electronique
DECUYPER Marcel	Géométrie
DELATTRE Charles	Géologie Générale
DELHAYE Michel	Chimie Physique
DERCOURT Michel	Géologie Générale
DURCHON Maurice	Biologie Expérimentale
FAURE Robert	Mécanique
FOURET René	Physique du Solide
GABILLARD Robert	Electronique
GLACET Charles	Chimie Organique
GONTIER Gérard	Mécanique
GRUSON Laurent	Algèbre
GUILLAUME Jean	Microbiologie
HEUBEL Joseph	Chimie Minérale
LABLACHE-COMBIER Alain	Chimie Organique
LANSRAUX Guy	Physique Atomique et Moléculaire
LAVEINE Jean-Pierre	Paléontologie
LEBRUN André	Electronique
LEHMANN Daniel	Géométrie

Mme	LENOBLE Jacqueline	Physique Atomique et Moléculai
M.	LINDER Robert	Biologie et Physiologie Végéta
M.	LOMBARD Jacques	Sociologie
M.	LOUCHEUX Claude	Chimie Physique
M.	LUCQUIN Michel	Chimie Physique
M.	MAILLET Pierre	Sciences Economiques
M.	MONTARIOL Frédéric	Chimie Appliquée
M.	MONTREUIL Jean	Biochimie
M.	PARREAU Michel	Analyse
M.	POUZET Pierre	Analyse Numérique
M.	PROUVOST Jean	Minéralogie
M.	SALMER Georges	Electronique
M.	SCHILTZ René	Physique Atomique et Moléculai
Mme	SCHWARTZ Marie-Hélène	Géométrie
M.	SEGUIER Guy	Electrotechnique
M.	TILLIEU Jacques	Physique Théorique
M.	TRIDOT Gabriel	Chimie Appliquée
M.	VIDAL Pierre	Automatique
M.	VIVIER Emile	Biologie Cellulaire
M.	WERTHEIMER Raymond	Physique Atomique et Moléculai
M.	ZEYTOUNIAN Radyadour	Mécanique

PROFESSEURS SANS CHAIRE

M.	BELLET Jean	Physique Atomique et Molécula
M.	BODARD Marcel	Biologie Végétale
M.	BOILLET Pierre	Physique Atomique et Molécula
M.	BOILLY Bénoni	Biologie Animale
M.	BRIDOUX Michel	Chimie Physique
M.	CAPURON Alfred	Biologie Animale
M.	CORTOIS Jean	Physique Nucléaire et Corpusc
M.	DEBOURSE Jean-Pierre	Gestion des entreprises
M.	DEPREZ Gilbert	Physique Théorique
M.	DEVRAINNE Pierre	Chimie Minérale
M.	GOUDMAND Pierre	Chimie Physique
M.	GUILBAULT Pierre	Physiologie Animale
M.	LACOSTE Louis	Biologie Végétale
Mme	LEHMANN Josiane	Analyse
M.	LENTACKER Firmin	Géographie
M.	LOUAGE Francis	Electronique
Mle	MARQUET Simone	Probabilités
M.	MIGEON Michel	Chimie Physique
M.	MONTEL Marc	Physique du Solide
M.	PANET Marius	Electrotechnique
M.	RACZY Ladislas	Electronique
M.	ROUSSEAU Jean-Paul	Physiologie Animale
M.	SLIWA Henri	Chimie Organique

MAITRES DE CONFERENCES (et chargés d'Enseignement)

M.	ADAM Michel	Sciences Economiques
M.	ANTOINE Philippe	Analyse
M.	BART André	Biologie Animale
M.	BEGUIN Paul	Mécanique
M.	BKOUCHE Rudolphe	Algèbre
M.	BONNELLE Jean-Pierre	Chimie
M.	BONNEMAIN Jean-Louis	Biologie Végétale
M.	BOSCQ Denis	Probabilités
M.	BREZINSKI Claude	Analyse Numérique
M.	BRUYELLE Pierre	Géographie

M. CARREZ Christian	Informatique
M. CORDONNIER Vincent	Informatique
M. COQUERY Jean-Marie	Psycho-Physiologie
Mle DACHARRY Monique	Géographie
M. DEBENEST Jean	Sciences Economiques
M. DEBRABANT Pierre	Géologie Appliquée
M. DE PARIS Jean-Claude	Mathématiques
M. DHAINAUT André	Biologie Animale
M. DELAUNAY Jean-Claude	Sciences Economiques
M. DERIEUX Jean-Claude	Microbiologie
M. DOUKHAN Jean-Claude	Physique du Solide
M. DUBOIS Henri	Physique
M. DYMENT Arthur	Mécanique
M. ESCAIG Bertrand	Physique du Solide
Me EVRARD Micheline	Chimie Appliquée
M. FONTAINE Jacques-Marie	Electronique
M. FOURNET Bernard	Biochimie
M. FORELICH Daniel	Chimie Physique
M. GAMBLIN André	Géographie
M. GOBLOT Rémi	Algèbre
M. GOSSELIN Gabriel	Sociologie
M. GRANELLE Jean-Jacques	Sciences Economiques
M. GUILLAUME Henri	Sciences Economiques
M. HECTOR Joseph	Géométrie
M. JACOB Gérard	Informatique
M. JOURNEL Gérard	Physique Atomique et Moléculaire
Mle KOSMAN Yvette	Géométrie
M. KREMBEL Jean	Biochimie
M. LAURENT François	Automatique
Mle LEGRAND Denise	Algèbre
Mle LEGRAND Solange	Algèbre
M. LEROY Jean-Marie	Chimie Appliquée
M. LEROY Yves	Electronique
M. LHENAFF René	Géographie
M. LOCQUENEUX Robert	Physique Théorique
M. LOUCHET Pierre	Sciences de l'Education
M. MACKE Bruno	Physique
M. MAHIEU Jean-Marie	Physique Atomique et Moléculaire
Me N'GUYEN VAN CHI Régine	Géographie
M. MAIZIERES Christian	Automatique
M. MALAUSSENA Jean-Louis	Sciences Economiques
M. MESSELYN Jean	Physique Atomique et Moléculaire
M. MONTUELLE Bernard	Biologique Appliquée
M. NICOLE Jacques	Chimie Appliquée
M. PAQUET Jacques	Géologie Générale
M. PARSY Fernand	Mécanique
M. PECQUE Marcel	Chimie Physique
M. PERROT Pierre	Chimie Appliquée
M. PERTUZON Emile	Physiologie Animale
M. PONSOLLE Louis	Chimie Physique
M. POVY Lucien	Automatique
M. RICHARD Alain	Biologie
M. ROGALSKI Marc	Analyse
M. ROY Jean-Claude	Psycho-Physiologie
M. SIMON Michel	Sociologie
M. SOMME Jean	Géographie
Mle SPIK Geneviève	Biochimie
M. STANKIEWICZ François	Sciences Economiques
M. STEEN Jean-Pierre	Informatique

.../...

M.	THERY Pierre	Electronique
M.	TOULOTTE Jean-Marc	Automatique
M.	TREANTON Jean-René	Sociologie
M.	VANDORPE Bernard	Chimie Minérale
M.	VILLETTE Michel	Mécanique
M.	WALLART Francis	Chimie
M.	WERNIER Georges	Informatique
M.	WATERLOT Michel	Géologie Générale
Mme	ZINN-JUSTIN Nicole	Algèbre

Monsieur le Professeur POUZET me fait l'honneur de présider le jury. Je le prie de croire en ma profonde gratitude.

Que Monsieur le Professeur HUARD, dont les conseils m'ont été précieux, et qui m'a toujours réservé un accueil bienveillant, veuille bien trouver ici l'expression de toute ma reconnaissance. Ce travail est un témoignage de plus, si besoin est, de son activité à l'Université de Lille.

Monsieur le Professeur BREZINSKI a accepté de juger ce travail, qu'il trouve ici l'expression de mes remerciements les plus sincères.

Monsieur le Professeur HANSEN par ses remarques pertinentes a amené à développer tout un chapitre de ce travail, qu'il accepte de croire en ma sincère reconnaissance.

Monsieur le Professeur LAPLACE a grandement facilité la réalisation de cette étude, tant par ses connaissances en informatique que par les facilités mises à ma disposition à l'I.U.T., qu'il trouve ici mes remerciements les plus sincères.

J'adresse mes remerciements les plus vifs à Jean-Charles FIOROT pour ses critiques franches et cordiales qui m'ont été très utiles.

La liste de mes créanciers est encore longue, je ne voudrais toutefois pas oublier,

Monsieur le Professeur VIGNES qui lors d'un stage, déjà lointain, à l'Institut Français du Pétrole a su me passionner pour les problèmes algorithmiques,

Monsieur le Professeur SPIELBERG qui nous a toujours encouragés et incités à persévérer,

Monique GUIGNARD-SPIELGERG qui a suivi ce travail avec intérêt et n'a cessé de nous conseiller.

Il serait injuste de ne pas évoquer mes horaires fantaisistes, mes humeurs maussades, les listings etc ..., toutes choses que ma famille a supportées avec patience et bonne humeur.

Je n'oublie pas Mademoiselle Bénédicte FIEVET ainsi que Monsieur et Madame DEBOCK qui ont permis l'excellente réalisation de ce mémoire en des délais particulièrement rapides.

" HÉRÉTIQUE ENTRE LES HÉRÉTIQUES
TOUJOURS M'ATTIRÈRENT
LES OPINIONS ÉCARTÉES "

A. GIDE

AVERTISSEMENT

En programmation mathématique et peut être plus encore en optimisation discrète, la construction de méthodes performantes de résolution de problèmes (dont l'origine concrète induit des tailles importantes) nécessite l'association de connaissances en analyse numérique et en informatique, d'autant plus qu'il existe de nombreuses interactions entre la modélisation d'un problème, la conception d'un algorithme et l'élaboration du programme associé.

D'autre part, toute analyse théorique doit s'accompagner d'une analyse expérimentale car il convient toujours d'évaluer les résultats au regard de ceux des méthodes déjà connues, ce qui entraine une multiplicité des expérimentations à réaliser, celles-ci dépassant largement le cadre du travail d'une seule personne puisqu'il est bien connu que les programmes dans ce domaine sont de grande dimension.

Toutes ces raisons, associées à un intérêt commun pour la programmation mathématique en nombres entiers, expliquent notre étroite collaboration qui a permis non seulement de très vite dépasser le cadre de la programmation pour arriver au stade théorique (sans jamais perdre de vue les implications pratiques), mais encore de pouvoir ainsi suivre plus facilement l'évolution extrêmement rapide de la discipline.

Comment par ailleurs, limiter un travail en commun effectif à des concertations épisodiques ? Chaque idée, au contraire, une fois émise, doit, pour être acceptée, avoir résisté aux remarques et aux critiques de l'autre qui l'enrichit de ses découvertes personnelles.

C'est dans cet esprit que nous nous sommes toujours efforcés de collaborer, aussi serait-il vain d'attribuer à l'un ou l'autre tel ou tel résultat. Puisque nous avons mis au point ensemble la totalité du travail, il nous est apparu que cette solidarité constante nous contraignait d'adopter une présentation commune.

" EFFICACE NE S'OPPOSE PAS À THÉORIQUE
MAIS À L'OPPOSITION
DU THÉORIQUE ET DU PRATIQUE "

R. DEBRAY

TABLE DES MATIERES

INTRODUCTION

NOTATIONS

GENERALITES (CHAPITRE 1)

INTRODUCTION 1

I - RELAXATIONS 2

I.1 - DÉFINITIONS ET RÉSULTATS GÉNÉRAUX 3

I.2 - COMPARAISON DE RELAXATIONS 6

I.3 - UTILISATION DES RELAXATIONS 11

II - COMPLEXITE DES ALGORITHMES 20

II.1 - NOTIONS GÉNÉRALES 21

II.2 - PROBLÈMES NP-COMPLETS 23

II.2.1 - Algorithmes non déterministes et problèmes de décision 25

II.2.2 - Problèmes Ardus 28

II.2.3 - Méthodes approchantes 30

LE PROBLEME DU KNAPSACK EN VARIABLES 0-1 (CHAPITRE 2)

INTRODUCTION 33

I - INTERET DU PROBLEME 37

I.1 - RELAXATIONS LAGRANGIENNES DU PROBLÈME D'AFFECTATION GÉNÉRALISÉ 37

I.2 - LE PROBLÈME DE DÉCOUPE À UNE DIMENSION

I.2.1 - Enoncé 40

I.2.2 - Formulation 41

I.2.3 - Utilisation du problème du knapsack 41

I.3 - ROTATION DE CONTRAINTES 43

I.4 - CONTRACTION DE CONTRAINTES

I.4.1 - Introduction 45

I.4.2 - Principe des méthodes 46

II - RELAXATION DU KNAPSACK 53

III - METHODES D'ENCADREMENT

III.1 - MINORANTS DE v(B) 60

III.1.1 - Algorithmes gloutons 61

III.1.2 - Algorithmes approchants polynômiaux 70

III.1.3 - Algorithmes approchants totalement polynômiaux 77

III.2 - MAJORANTS DE v(B)

III.2.1 - Relaxation du problème asymptotique de Gomory 89

III.2.2 - Relaxations de $(B \mid x_i = 0)$ et $(B \mid x_i = 1)$ 90

III.3 - MINORANT DE AX^* 91

III.4 - ENCADREMENT DE Σx_j 93

IV - RELAXATION LAGRANGIENNE

IV.1 - RÉSULTATS GÉNÉRAUX 97

IV.2 - RELAXATIONS LAGRANGIENNES PRIVILÉGIÉES 99

IV.3 - MEILLEURES RELAXATIONS LAGRANGIENNES 100

V - REDUCTION

V.1 - SYNTHÈSE 107

V.2 - RÉSULTATS PRÉLIMINAIRES

V.2.1 - Relations de dominance 111

V.2.2 - Relations binaires 116

V.3 - NOUVEL ALGORITHME DE RÉDUCTION 118

VI - ENUMERATION IMPLICITE 125

VII - EXPERIENCES NUMERIQUES 138

VII.1 - RELAXATION

VII.1.1 - Mise en oeuvre 140

VII.1.2 - Résultats numériques 141

VII.2 - RÉDUCTION 151

VII.3 - ALGORITHME MSB 79

VII.3.1 - Analyse des résultats de la méthode MSB 71 159

VII.3.2 - Résultats de la méthode MSB 79 163

PROBLEMES EN NOMBRES ENTIERS A PLUSIEURS CONTRAINTES (CHAPITRE 3)

INTRODUCTION 165

I - REDUCTION POUR LES PROBLEMES A PLUSIEURS CONTRAINTES

INTRODUCTION 168

I.1 - FORMULATION DU PROBLÈME 168

I.2 - TESTS DE RÉDUCTION DE LA TAILLE DE (B) 169

I.2.1 - Elimination de variables 169

I.2.2 - Elimination de contraintes 171

I.3 - ALGORITHME DE RÉDUCTION 173

I.4 - EXPÉRIENCES NUMÉRIQUES 176

II - RESOLUTION D'UN PROBLEME D'AFFECTATION

INTRODUCTION 181

II.1 - PROBLÈME ÉTUDIÉ

II.1.1 - Enoncé 182

II.1.2 - Formulation 182

II.1.3 - Propriétés relatives aux seconds membres des contraintes 185

II.2 - RÉSOLUTION DU PROBLÈME (PB)

II.2.1 - Modification de la formulation du problème 187

II.2.2 - Description de la méthode 188

II.3 - EXPÉRIENCES NUMÉRIQUES 198

III - RESOLUTION D'UN PROBLEME DE FABRICATION

III.1 - PROBLÈME ÉTUDIÉ

III.1.1 - Enoncé 202

III.1.2 - Formulation 202

III.2 - RÉSOLUTION DE (PF)

III.2.1 - Réduction "triviale" de la taille de (PF) 205

III.2.2 - Minorant de v(PF) 205

III.2.3 - Résolution de $(\overline{PF})$ 207

III.2.4 - Réduction de la taille de (PF) 208

III.2.5 - Schéma exploratoire 208

III.3 - EXPÉRIENCES NUMÉRIQUES 209

UN TYPE DE PROBLEME DU KNAPSACK NON LINEAIRE (CHAPITRE 4)

I - INTRODUCTION 211

II - NOTATIONS 214

III - RESULTATS PRELIMINAIRES

III.1 - ENCADREMENT DE $\sum_J x_J$ 215

III.2 - LINÉARISATIONS DU PROBLÈME - APPLICATIONS 222

IV - METHODE DE REDUCTION

IV.1 - PRINCIPE DE LA MÉTHODE 230

IV.2 - METHODES D'ENCADREMENT DE $\sum_J x_J$ 232

IV.3 - POINTS EXTREMES DE L'ENVELOPPE CONVEXE D'UN ENSEMBLE DE POINTS DE Z^2 238

IV.4 - ALGORITHMES GLOUTONS POUR LES BORNES INFÉRIEURES À v(K) 260

IV.5 - ALGORITHME D'ÉLIMINATION DÉFINITIVE DE VARIABLES 282

IV.6 - EXPÉRIENCES NUMÉRIQUES 287

V - EXPERIENCES NUMERIQUES 294

ANNEXE : DONNEES DE PROBLEMES TESTS 308

BIBLIOGRAPHIE

CODE DE L'ALGORITHME MSB 79

INTRODUCTION

Ce travail s'inscrit dans le cadre des études consacrées à la programmation mathématique en nombres entiers dont les applications sont répandues dans les domaines économiques et techniques (problèmes d'investissement, de localisation d'usines, d'implantations d'entrepôts, d'affectation de personnel, d'utilisation de machines, de découpes, etc...).

Il concerne plus précisément la résolution des programmes mathématiques à n variables bivalentes*, c'est-à-dire des problèmes d'optimisation sur le cube unité de $\mathbb{R}^n$ d'une fonctionnelle linéaire soumise à des contraintes linéaires ou non exprimées sous forme d'égalités ou d'inégalités.

Cet exposé n'a pas pour but d'énoncer la méthode de résolution universelle mais plutôt de contribuer à son élaboration (si jamais elle existe un jour !) en faisant état de nos résultats aussi bien théoriques qu'expérimentaux obtenus depuis quelques années.

Il est scindé en quatre chapitres : l'exposé des généralités (chapitre 1) utilisées dans les autres chapitres est suivi de l'étude approfondie du problème de base, appelé problème du knapsack (qui ne possède qu'une seule contrainte) (chapitre 2) et de son utilisation dans le contexte des problèmes à plusieurs contraintes (chapitre 3). Le dernier chapitre concerne la résolution d'un type de problème du knapsack non linéaire appelé "knapsack baudruche".

Le développement de chaque chapitre est précédé d'une introduction qui le situe dans l'ensemble de l'exposé et en précise la portée.

Le chapitre 1 est consacré à l'énoncé des concepts généraux suivants :

---

* il est toujours possible d'envisager des variables ne prenant que les valeurs 0 ou 1 par simple changement de variables.

La partie I fournit les différents types de stratégies pouvant être employées pour accélérer la résolution du problème tout en le transformant en un problème plus simple : les définitions et résultats généraux relatifs aux relaxations, relaxations lagrangiennes, contraintes composites, y précèdent la description de leurs utilisations (obtention de bornes, réduction de la taille du problème) dans les méthodes dites d'énumération implicite qui constituent l'une des deux grandes classes à laquelle nous avons consacré nos recherches (seuls les principes et inconvénients des méthodes dites algébriques, formant l'autre classe, seront rappelés).

Les problèmes de convergence des algorithmes ne se posent pas en programmation mathématique en nombres entiers puisque le nombre des solutions de tout problème est fini. Mais ce nombre pouvant atteindre 2^n(nombre de sommets du cube unité de $\mathbb{R}^n$) et devenir ainsi démesurément grand, même pour les petites tailles de problèmes (par exemple, 2^{50} est supérieur à 10^{15}), il convient d'analyser la complexité des algorithmes de résolution, c'est-à-dire de pouvoir mesurer théoriquement les coûts des algorithmes en temps d'exécution et en espace mémoire nécessaire à leur implantation sur ordinateur. C'est l'objet de la partie II qui fournit, après les notions générales sur la compléxité théorique, un résumé sur la théorie des problèmes réputés difficiles (NP-complet, Ardu) débouchant sur les définitions relatives aux méthodes dites approchantes qui aboutissent à un minorant de la valeur du problème avec une précision donnée a priori.

Le chapitre 2 est une étude exhaustive du problème fondamental en optimisation discrète : le problème dit "du knapsack" en variables bivalentes, qui possède une contrainte unique en inégalité :

Son intérêt indéniable dans le cadre de la résolution des problèmes concrets de grandes tailles (partie I : relaxation lagrangienne, découpe, rotation et contraction de contraintes) a conduit de nombreux auteurs à élaborer des méthodes de résolution du knapsack appartenant à deux grandes classes :

- les méthodes approchantes : dont une synthèse est réalisée au paragraphe III.1 (pour aussi simple que soit sa formulation, le knapsack fait partie des problèmes d'optimisation réputés les plus difficiles)

- les méthodes exactes : la portée de la programmation dynamique (rappelée au paragraphe III.1) étant limitée par la taille des problèmes et la grandeur de leurs données, et les cas les plus défavorables exhibés en théorie n'apparaissant pas en général dans la pratique, une majorité d'auteurs ont mis en oeuvre des algorithmes d'énumération implicite aboutissant à la solution exacte du problème en des temps d'exécution très acceptables.

Il est actuellement clair que les plus efficaces d'entre eux comportent trois phases successives :

Phase 1 : résolution de la relaxation

Phase 2 : réduction de la taille (élimination de variables après attribution de valeurs bien définies)

Phase 3 : résolution du problème réduit par une méthode d'énumération implicite.

Cette dernière phase n'occupant qu'une place infime, du point de vue temps de calculs, par rapport aux deux premières - par exemple, pour les problèmes de 5000 variables, les temps requis pour la phase 3 sont de l'ordre de 7.4 % des temps globaux - nous nous sommes uniquement attachés à rappeler en partie VI deux principes d'énumération, après avoir décrit la succession des tests permettant de parcourir implicitement l'arborescence de l'ensemble des solutions.

Par contre, notre intérêt s'est porté plus particulièrement sur l'étude approfondie des deux premières phases pour lesquelles nous proposons de faire le point en apportant à chaque fois notre contribution :

La méthode classique de résolution de la relaxation du knapsack comporte un classement initial des données dont la part énorme des temps de calcul par rapport aux temps globaux de résolution a conduit à la mise au point d'une nouvelle méthode très efficace ne nécessitant plus ce tri systématique des données (partie II) ; à titre d'exemple, cette méthode de complexité linéaire aboutit à un gain de l'ordre de 74 % pour les problèmes de 9000 variables.

L'exposé d'un catalogue de méthodes d'encadrement sur la fonction économique et sur la contrainte (partie III), et de résultats théoriques relatifs aux relaxations lagrangiennes du knapsack (partie IV) précède la présentation d'un cadre commun pour tous les algorithmes de réduction connus, et d'une nouvelle méthode performante (quant aux nombres de variables éliminées) mettant en jeu des relaxations, des relaxations lagrangiennes, et des relations binaires et de dominance entre variables (partie V).

Enfin la partie VII regroupe les expériences numériques associées aux algorithmes présentés en parties II, V et VI ; elles ont été réalisées avec plus de 550 problèmes tests (450 tirés au hasard suivant la loi uniforme de 50 à 13000 variables ; les autres (jusqu'à 200 variables) étant extrai de la littérature). Les résultats relatifs à une nouvelle version de notre méthode de résolution exacte y sont en particulier répertoriés ; ils montrent que cette méthode - appelée MSB 79 - est de complexité expérimentale linéaire et plus rapide que celle de Martello et Toth - appelée MT - (publiée tout récemment) qui était annoncée par les auteurs comme la plus performante [par exemple, pour n = 5000, la méthode MSB 79 est près de 3 fois plus rapide que la méthode MT ; même en excluant les temps de la phase 1, le gain en temps d'exécution apporté par notre méthode est de l'ordre de 22 %].

Le chapitre 3 concerne les problèmes en variables entières à plusieurs contraintes (tous les exemples envisagés sont d'origine concrète) le contenu de chacune de ses trois parties est une nouvelle preuve de l'importance de l'étude du chapitre 2 :

La première partie montre que la réduction (à la fois en nombre de variables et en nombre de contraintes) d'un problème à plusieurs contraintes est réalisée efficacement par la conjonction de tests simples appliqués en particulier sur des problèmes du knapsack extraits du problème d'origine [les problèmes d'investissement traités sont tirés de la littérature ; la taille maximale envisagée est : 5 contraintes, 50 variables].

Les deux autres parties ont pour objet les résolutions de deux types de problèmes qui nous ont été personnellement proposés :

(i) un problème d'affectation (implantation de groupes de divers sites sur plusieurs périodes) - partie II - dont la particularité est de posséder des contraintes du type partitionnement, l'ensemble S des autres contraintes n'admettant pour éléments non nuls que des coefficients égaux à 1. Sa transformation en un problème du knapsack équivalent, par contraction des contraintes de S en une contrainte unique (dont l'ensemble des solutions entières réalisables est identique à celui du système) a permis sa résolution par une adaptation de la méthode exposée au chapitre 2. Les expériences numériques montrent que les temps d'exécution sont comparables à ceux relatifs aux problèmes du knapsack tirés au hasard [taille maximale envisagée : 14 contraintes, 160 variables ; temps de résolution associé : 59 secondes sur Gamma M 40].

(ii) un problème de fabrication qui se pose dans de nombreuses branches industrielles, en particulier dans le domaine de la fonderie lorsqu'il s'agit de regrouper des commandes de tôles dans le but de réaliser les coulées de métal au meilleur coût (partie III). Sa résolution comporte entre autres :

- une élimination de contraintes et de variables basée non seulement sur la relaxation lagrangienne, mais également sur les aspects économiques du problème

- un algorithme glouton s'appuyant sur des résolutions exactes de problèmes du knapsack issus du problème initial, et fournissant très rapidement de bons minorants des valeurs des exemples traités.

[taille maximale envisagée : 174 contraintes, 1038 variables ; temps d'exécution associé : 162 secondes sur UNIVAC 1110].

L'exposé s'achève par l'étude d'un type de problème du knapsack non linéaire dont le second membre de la contrainte est une fonction non croissante de la somme des variables (chapitre 4) :

Ce problème, introduit par Guignard et Posner [G.P.] est également d'origine concrète : parmi les diverses interprétations économiques, citons d'une part, la généralisation de celle du knapsack classique lorsque le conditionnement des objets mis dans le sac, est pris en compte, d'autre part, l'exploitation optimale d'un ordinateur en multiprogrammation.

Une méthode de réduction de la taille de ce problème appelé "knapsack baudruche" (en anglais, "collapsing knapsack") présentée en partie IV, inclut :

(i) la détermination d'un encadrement de la somme des variables en utilisant des généralisations de l'algorithme pour la relaxation du knapsack proposé en partie II du chapitre 2.

(ii) la construction d'une enveloppe convexe de point de $\mathbb{Z}^2$ qui est à la base de la détermination de deux types de problèmes du knapsack classique : le premier permet la génération d'un algorithme glouton pour une borne inférieure de la valeur du problème ; la résolution de la relaxation du second fournit une borne supérieure de la valeur du problème et conduit à un schéma de réduction utilisant la relaxation lagrangienne et la contraction de contraintes.

Les résultats numériques de la partie V montrent que la méthode proposée donne des résultats satisfaisants quant aux nombres de variables éliminées et à l'évolution des temps de calcul avec la taille des problèmes :

- elle suffit à elle seule dans de nombreux cas - et en particulier pour les cinq problèmes tests de [G.P.] - pour fournir soit la solution optimale, soit un sous-problème dont la solution sera obtenue par une technique simple d'énumération.

- l'apport de cette réduction en phase initiale de la méthode proposée par [G.P.] permet une accélération importante de la résolution : les temps d'exécution sont couramment divisés par 100 et plus.

Les résultats présentés regroupent pour l'essentiel différentes publications effectuées depuis 1972. Toutefois, il nous a semblé préférable de les remodeler et les présenter avec une certaine unité ; ce qui a permis d'inclure les résultats les plus récents non encore publiés. Cet exposé ne doit pas être considéré comme une fin en soi, les différents chapitres offrant encore de nombreuses possibilités de développement.

NOTATIONS

Les notations utilisées tout au long de l'exposé sont les suivantes :

$\lfloor \lambda \rfloor$: partie entière d'un réel λ ; $\lceil \lambda \rceil = - \lfloor -\lambda \rfloor$

étant donné un ensemble J :

$[J]$: enveloppe convexe de J

$|J|$: cardinal de J

$J \setminus U$: complémentaire d'un sous-ensemble U de J

étant donné A *une matrice de format* $(I \times J)$ *où* I *et* J *sont des ensembles finis :*

a_{ij} : élément $(i,j) \in I \times J$ de A

A^j : colonne $j \in J$ de A

A_i : ligne $i \in I$ de A

$A^{J'}$: sous-matrice de A composée des colonnes A^j $j \in J' \subset J$

$A_{I'}$: sous-matrice de A composée des lignes A_i $i \in I' \subset I$

étant donné y *un vecteur (ligne ou colonne) indicé par un ensemble* J :

y_j : élément $j \in J$ de y

$y_{J'}$: sous-vecteur de y composé des éléments y_j $j \in J' \subset J$

étant donné un problème d'optimisation (P) :

$v(P)$: valeur optimale de (P)

$\bar{v}(P)$: (resp. $\underline{v}(P)$) borne supérieure (resp. inférieure) de $v(P)$

$F(P)$: domaine de (P)

$(P \mid x \in X)$: lorsque (P) est résolu avec les conditions supplémentaires $x \in X$ [i.e. $(\exists j : x_j = \varepsilon)$ <u>ou</u> $(\exists j_1, j_2 : x_{j_1} = \varepsilon_1 \; x_{j_2} = \varepsilon_2)$ <u>ou</u> $(\sum_j x_j \geq \ell)$ avec $\varepsilon, \varepsilon_1, \varepsilon_2 \in \{0,1\}$, $\ell \in \mathbb{N}$]

$x_j \leftarrow \varepsilon$: x_j est fixée à la valeur $\varepsilon \in \{0,1\}$

lorsque (P) *est un problème à* n *variables bivalentes :*

x^* : solution optimale de (P)

$V = \{x \in \mathbb{R}^n \mid x_j = 0 \text{ ou } 1, \; j = 1,\ldots,n\}$

$(\bar{P})$: problème (P) dans lequel $[V]$ remplace V

$\bar{x}$: solution optimale de $(\bar{P})$

étant donné $x \in [V]$:

$[x]$: élément de V tel que $[x]_j = \lfloor x_j \rfloor$ $j = 1,\ldots,n$

GENERALITES

CHAPITRE 1

INTRODUCTION

Ce chapitre est consacré à l'énoncé de concepts généraux qui entreront en ligne de compte dans les études des chapitres suivants.

Deux questions fondamentales liées à la programmation mathématique en variables bivalentes sont abordées :

* *Quels types de stratégies peut-on employer pour transformer le problème d'origine en un problème plus simple fournissant des informations conséquentes pour accélérer la résolution du problème initial ?*

La description de l'utilisation des relaxations (types lagrangiennes ou contraintes composites) répondra à cette question.

* *Comment évaluer le coût d'une méthode de résolution ?*

Ce sera l'objet de l'analyse de la complexité des algorithmes.

I - RELAXATIONS

Dans la plupart des problèmes de programmation mathématique en nombres entiers, c'est uniquement un sous-ensemble S de contraintes (en anglais "side constraints") qui rendent leurs résolutions "ardues". Deux grands principes simples (mais efficaces comme il sera vu dans les chapitres futurs) permettent de générer un nouveau problème qui

- d'une part, sera *plus facile à résoudre que le problème initial.*

- d'autre part, fournira une *borne* supérieure (resp. inférieure) de la valeur du problème d'origine, si celui-ci est un problème de maximisation (resp. minimisation).

Principe 1 : Combiner linéairement les |S| contraintes pour aboutir à une contrainte unique dite "composite".

Principe 2 : "Supprimer" les |S| contraintes

- soit en les éliminant du problème ; cela générera (comme dans le principe 1) ce qu'on appelera au sens large une *relaxation.*
- soit en les introduisant par combinaison linéaire dans la fonction économique (le terme "dualisation" sera également employé) pour aboutir à une *relaxation lagrangienne.*

Note :

Il est possible d'imaginer une stratégie hybride faisant intervenir à la fois les deux principes précédemment évoqués, c'est-à-dire par exemple partitionner S en S_1, S_2, S_3 dans le but de

- transformer les contraintes de S_1 en contrainte composite
- éliminer les contraintes de S_2
- dualiser les contraintes de S_3.

□

Les définitions et résultats généraux relatifs aux relaxations et relaxations lagrangiennes (§ I.1. et I.2.) précédent la description de leurs utilisations dans le cadre des méthodes dites d'énumération implicite (§ I.3.).

I.1 - DÉFINITIONS ET RÉSULTATS GÉNÉRAUX

Définition 1

Un problème de maximisation (resp. minimisation) (Q) est appelé relaxation d'un problème de maximisation (resp. minimisation) (P) si

(i) $F(Q) \supseteq F(P)$

(ii) *la fonction économique de (Q) majore (resp. minore) celle de (P) sur F(P)*

Note :

$(\bar{P})$ est un exemple de relaxation de (P).

□

Définition 2

(i) *Etant donné un problème de maximisation du type suivant :*

$$(P) \quad \begin{cases} \text{maximiser} & cx \\ \text{sous les conditions} & Ax \leq b \\ & Cx \leq d \\ & x \in V \end{cases}$$

On appelle relaxation lagrangienne de (P) associée aux contraintes $Ax \leq b$ *et à* $u \geq 0$ *(vecteur ligne à composantes réelles de même dimension que b), le problème*

$$(RP(u)) \quad \begin{cases} \max\limits_{x} & cx + u(b - Ax) \\ \text{s.c.} & Cx \leq d \\ & x \in V \end{cases}$$

(ii) *Le dual de (P) est le problème de minimisation*

$$(D) \quad \text{minimiser } v(RP(u)) \quad \text{s.c. } u \geq 0$$

Définition 2 bis

(i) *Etant donné un problème de minimisation du type suivant :*

$$(P) \quad \begin{cases} \min \ cx \\ \text{s.c. } Ax \geq b \\ \quad\ Cx \geq d \\ \quad\ x \in V \end{cases}$$

On appelle relaxation lagrangienne de (P) associée aux contraintes $Ax \geq b$ *et à* $u \geq 0$ *(vecteur ligne à composantes réelles de même dimension que b), le problème*

$$(RP(u)) \quad \begin{cases} \min\limits_{x} \ cx + u(b - Ax) \\ \text{s.c. } Cx \geq d \\ \quad\ x \in V \end{cases}$$

(ii) *Le dual de (P) est le problème de maximisation*

$$(D) \quad \begin{cases} \max \ v(RP(u)) \\ \text{s.c. } u \geq 0 \end{cases}$$

Note :

(i) $\forall u \geq 0$ (RP(u)) est une relaxation de (P)

(ii) le domaine de (RP(u)) étant indépendant de u sera noté par la suite F(RP).

□

Définition 3

La relaxation lagrangienne d'un problème d'optimisation (P) possède la propriété d'intégralité si $v(RP(u)) = v(\overline{RP(u)})$ $\forall u \geq 0$.

Exemples : dans le cas particulier où les contraintes $Cx \leq d$ (ou $Cx \geq d$) n'existent pas (i.e. toutes les contraintes sont "dualisées"), alors

(i) si (P) est un problème de maximisation

$$v(\overline{RP(u)}) = v(RP(u)) = ub + \max \{(c - uA)\, x \mid x \in V\}$$
$$= ub + \sum_{j \in J^{+}(u)} (c_j - uA^j)$$

où $J^{+}(u) = \{j \in \{1,\ldots,n\} \mid c_j - uA^j > 0\}$.

(ii) si (P) est un problème de minimisation

$$v(\overline{RP(u)}) = v(RP(u)) = ub + \min \{(c - uA)\, x \mid x \in V\}$$
$$= ub + \sum_{j \in J^{-}(u)} (c_j - uA^j)$$

où $J^{-}(u) = \{j \in \{1,\ldots,n\} \mid c_j - uA^j < 0\}$.

□

Les résultats suivants démontrés dans Geoffrion [G23] montrent que la mesure du "gap" (ou valeur absolue de l'écart entre v(P) et v(D)) est fournie exactement par $| v(\overline{P}) - v(P) |$ lorsque (RP(u)) ($u \geq 0$) possède la propriété d'intégralité. Lorsque cette propriété n'est pas vérifiée, un multiplicateur $u \geq 0$ pourra être tel que :

$$| v(P) - v(RP(u)) | < | v(P) - v(\overline{P}) |$$

Théorème 1

En notant $\overline{u}$ *le multiplicateur associé au système* $Ax \leq b$ *à l'optimum de* $(\overline{P})$

Si $F(P) \neq \emptyset$ *alors*

(i) *si (P) est un problème de maximisation*

$$v(P) \leq v(D) = \max \{cx \mid Ax \leq b \; ; \; x \in [(F(RP)]\}$$
$$\leq v(RP(\overline{u})) \leq v(\overline{P})$$

(ii) *si (P) est un problème de minimisation*

$$v(P) \geq v(D) = \min \{cx \mid Ax \geq b \; ; \; x \in [(F(RP)]\}$$
$$\geq v(RP(\overline{u})) \geq v(\overline{P}).$$

Théorème 2

Lorsque la relaxation lagrangienne de (P) possède la propriété d'intégralité et $F(P) \neq \emptyset$ *alors* $v(D) = v(RP(\overline{u})) = v(\overline{P})$.

I.2 - COMPARAISON DE RELAXATIONS

Un autre exemple de relaxation d'un problème du type

$$(P) \quad \left[\begin{array}{l} \text{optimiser } cx \\ \quad \text{s.c. } Ax \; R \; b \\ \quad\quad Cx \; R \; d \\ \quad\quad x \in V \end{array}\right.$$

(où R représente la relation $\leq$ ou $\geq$)
est le problème obtenu en remplaçant un ensemble de contraintes (par exemple $Ax \; R \; b$) par une contrainte unique dite *composite* construite par combinaison linéaire (voir, par exemple, [G26]) i.e. :

$$(P(u)) \quad \left[\begin{array}{l} \text{optimiser } cx \\ \quad \text{s.c. } uAx \; R \; ub \\ \quad\quad Cx \; R \; d \\ \quad\quad x \in V \end{array}\right.$$

(avec $u \geq 0$ vecteur ligne à composantes réelles de même dimension que b).

L'étude suivante concerne un problème de maximisation de la forme

$$(P)\quad \begin{cases} \max\ cx \\ \text{s.c. } Ax \le b \\ \quad\ Cx \le d \\ \quad\ x \in V \end{cases}$$

et ses deux types de relaxations, $u \ge 0$ donné :

$$(P(u))\quad \begin{cases} \max\ cx \\ \text{s.c. } uAx \le ub \text{ (contrainte composite)} \\ \quad\ Cx \le d \\ \quad\ x \in V \end{cases}$$

$$(RP(u))\quad \begin{cases} \max\limits_{x}\ cx + u(b - Ax) \\ \text{s.c. } Cx \le d \qquad \text{(relaxation lagrangienne)} \\ \quad\ x \in V \end{cases}$$

Le théorème suivant montre que "le gap" fourni par une contrainte composite peut être inférieur à celui fourni par la relaxation lagrangienne :

Théorème 3

$$\min_{u\ge 0} v(P(u)) \le \min_{u\ge 0} v(RP(u))$$

Démonstration

$\forall u \ge 0$ $(RP(u))$ est la relaxation lagrangienne de $(P(u))$ associée à la contrainte $uAx \le ub$ et au multiplicateur égal à 1, d'où

$$\forall u \ge 0 \quad v(P(u)) \le v(RP(u))$$

En particulier, cette inégalité est vérifiée pour

$$\bar{u} : v(RP(\bar{u})) = \min_{u\ge 0} v(RP(u))$$

$$\Rightarrow \min_{u\ge 0} v(P(u)) \le v(P(\bar{u})) \le \min_{u\ge 0} v(RP(u))$$

Dans le cas particulier où les contraintes $Cx \leq d$ n'existent pas, (RP(u)) possédant la propriété d'intégralité, le résultat précédent se précise en montrant que

$$\min_{u \geq 0} v(RP(u)) = v(\bar{P}) :$$

Lemme

Etant donnés, un domaine convexe D de $\mathbb{R}^m \times \mathbb{R}^n$

$D_x = \{y \in \mathbb{R}^n \mid (x,y) \in D\} \quad \forall x \in \mathbb{R}^m$,
les fonctions f *et* g *suivantes*

$$f : \mathbb{R}^m \to \mathbb{R}$$

et $\forall x \in \mathbb{R}^m$

$$g : D_x \subset \mathbb{R}^n \to \bar{\mathbb{R}} \text{ tel que } g(y) = \begin{cases} +\infty & \text{si } D_x = \emptyset \\ r_{x,y} \in \mathbb{R} & \text{sinon } (y \in D_x) \end{cases}$$

$(\bar{x},\bar{y})$ *une solution optimale du problème :*

(P_1) $\min f(x) + g(y)$ *s.c.* $(x,y) \in D$,

et les problèmes :

(P_2) $\min_y f(\bar{x}) + g(y)$ *s.c.* $y \in D_{\bar{x}}$

(P_3) $\min_{x \in \mathbb{R}^m} [f(x) + \min g(y)$ *s.c.* $y \in D_x]$

alors les problèmes (P_1), (P_2), (P_3) *sont équivalents.*

Démonstration

1. $(P_1) \Longleftrightarrow (P_2)$

a) $f(\bar{x}) + g(\bar{y}) \leq f(x) + g(y) \quad \forall (x,y) \in D.$

Donc, en particulier :

$f(\bar{x}) + g(\bar{y}) \leq f(\bar{x}) + g(y)\ \forall\ (\bar{x},y) \in D$ (i.e. $\forall y \in D_{\bar{x}}$).

$$\Longrightarrow f(\bar{x}) + g(\bar{y}) \leq f(\bar{x}) + \min \{g(y) \mid y \in D_{\bar{x}}\} \qquad (1)$$

b) $\min \{g(y) \mid y \in D_{\bar{x}}\} \leq g(\bar{y})$ puisque $\bar{y} \in D_{\bar{x}}$

$$\Longrightarrow f(\bar{x}) + \min \{g(y) \mid y \in D_{\bar{x}}\} \leq f(\bar{x}) + g(\bar{y}) \qquad (2)$$

(1)et (2) $\Longrightarrow$ $\underline{f(\bar{x}) + g(\bar{y}) = f(\bar{x}) + \min \{g(y) \mid y \in D_{\bar{x}}\}}$

2. $\underline{(P_1) \Longleftrightarrow (P_3)}$:

a) $f(\bar{x}) + g(\bar{y}) \leq f(x) + g(y)\ \forall\ (x,y) \in D$ (i.e. $\forall y \in D_x$)

$$\leq f(x) + \min \{g(y) \mid y \in D_x\}$$

avec $\min\limits_{y \in D_x} g(y) = \begin{cases} +\infty \text{ si } D_x = \emptyset \\ r_x \in \mathbb{R} \text{ sinon} \end{cases}$

$$\Longrightarrow f(\bar{x}) + g(\bar{y}) \leq \min_{x \in \mathbb{R}^m} [f(x) + \min \{g(y) \mid y \in D_x\}] \qquad (3)$$

b) $z = \min\limits_{x \in \mathbb{R}^m} [f(x) + \min \{g(y) \mid y \in D_x\}]$

$$\leq f(x) + \min \{g(y) \mid y \in D_x\} \quad \forall x \in \mathbb{R}^m$$

en particulier :

$$z \leq f(\bar{x}) + \min \{g(y) \mid y \in D_{\bar{x}}\}$$

D'après la première partie de la démonstration, on peut donc conclure que

$$\min_{x \in \mathbb{R}^m} [f(x) + \min \{g(y) \mid y \in D_x\}] \leq f(\bar{x}) + g(\bar{y}) \qquad (4)$$

(3) et (4) $\Longrightarrow f(\bar{x}) + g(\bar{y}) = \min\limits_{x \in \mathbb{R}^m} [f(x) + \min \{g(y) \mid y \in D_x\}]$.

□

D'où le

Théorème 3 bis

Etant donnés les problèmes

$$(Q_1) \quad \min_{u \geq 0} [\max cx \text{ s.c. } uAx \leq ub \; ; \; x \in V]$$

et

$$(Q_2) \quad \min_{u \geq 0} [ub + \max_{x} (c - uA)x \text{ s.c. } x \in V]$$

alors $v(Q_1) \leq v(Q_2) = v(\bar{P})$

Démonstration

(i) $v(Q_1) \leq v(Q_2)$: conséquence directe du théorème 3

(ii) $v(Q_2) = v(\bar{P})$: $u \geq 0$ étant donné, le dual de

$$\max_{x} (c - uA)x \text{ s.c. } Ux \leq e \; ; \; x \geq 0$$

(où U désigne la matrice unité d'ordre n et e le vecteur colonne dont les n éléments valent 1) s'écrit

$$\min we \text{ s.c. } uA + w \geq c \; ; \; w \geq 0$$

(avec $w \in \mathbb{R}^n$ multiplicateur de Lagrange associé à $Ux \leq e$).

Le lemme permet alors de conclure (en posant $x = u$ et $y = w$) que (Q_2) est équivalent à

$$\min ub + we \text{ s.c. } uA + w \geq c \; ; \; u \geq 0 \; ; \; w \geq 0$$

dual de (P), d'où la conclusion.

□

I.3 - UTILISATION DES RELAXATIONS

Soit le problème

$$(B)\quad \begin{cases} \max\ cx \\ \text{s.c. } Ax \leq b \\ \qquad x \in V \end{cases}$$

où A est une matrice de format m x n
c est une ligne de format n ; b une colonne de format m
telles que $c_j, a_{ij}, b_i \in \mathbb{Z}$; $\forall i \in \{1,\ldots,m\}$, $\forall j \in \{1,\ldots,n\}$.

Il existe deux grandes classes de méthodes de résolution de (B) :
* les *méthodes algébriques* (dites "de coupe") dûes à Gomory [G29] dont le principe est la construction de l'enveloppe convexe de F(B), tout au moins au voisinage de x^* : si la solution de $(\bar{B})$ n'est pas entière - sinon le problème (B) est résolu - ces méthodes génèrent une séquence de contrainte additionnelles (appelées "plans de coupure") qui réduisent $F(\bar{B})$ tout en gardant intact le domaine entier, jusqu'à l'obtention d'un problème dont la solution (en continu) est une solution de (B) (voir [G4, G8])
i.e. en notant (B_p) le problème (B) avec p contraintes additionnelles
$\exists q \in \mathbb{N}$: les problèmes $(B_1), (B_2),\ldots,(B_q)$ sont résolus tour à tour de sorte que

$$F(\bar{B}) \supset F(\bar{B}_1) \supset F(\bar{B}_2) \supset \ldots \supset F(\bar{B}_q) \supset [F(B)]$$
$$\text{et } \max\{cx \mid x \in F(\bar{B}_q)\} = v(B).$$

Les nombreux inconvénients de ces méthodes :

- les hyperplans de coupure peuvent être pratiquement parallèles dans les dernières itérations, ce qui peut créer des phénomènes de dégénérescence.

- une *solution réalisable unique* est fournie en fin de méthode, d'où la nécessité de prévoir la sauvegarde des résultats associés à chaque problème du type $(\bar{B}_p)$

- la classe des problèmes pouvant être traités est très limitée ; tous les sous-déterminants de la matrice des contraintes ne devant pas être trop grands, ce sont essentiellement les problèmes de recouvrement et de partionnement qui vérifient ces conditions.
font que nous nous sommes particulièrement intéressés à la seconde classe ;

* les *méthodes d'énumération implicite* (encore appelées "Branch and Bound", Séparation et Evaluation Progressive ou Séquentielle), plus efficaces et réalistes que les précédentes. (Voir [G7,G12,G21,G33,G34,G35,G36,G43,G45,G46, G52,G55,G57,G58,G64]). Leurs deux principes fondamentaux sont :

(i) définir un schéma d'exploration exhaustive de l'ensemble des solutions, c'est-à-dire en fait l'ensemble V des 2^n sommets du cube unité (c'est la notion de *séparation* qui permet de décrire V sous forme d'arborescence).

(ii) en chaque noeud de cette arborescence, c'est-à-dire lorsque un sous-ensemble de variables ont été fixées soit à 0 (ensemble d'indices noté X_0) soit à 1 (ensemble d'indices noté X_1), considérer le sous-problème ainsi généré :

$$(P) \equiv (B \mid x_j = 1 \ \forall j \in X_1 \ ; \ x_j = 0 \ \forall j \in X_0)$$

$$\text{i.e.} \left[\begin{array}{l} \sum_{j \in X_1} c_j + \max \sum_{j \in X_2} c_j x_j \\ \qquad \text{s.c.} \sum_{j \in X_2} A^j x_j \leq b - \sum_{j \in X_1} A^j \\ \qquad\quad x_j = 0 \text{ ou } 1 \quad \forall j \in X_2 \end{array} \right.$$

où $X_2 = \{1,\ldots,n\} \setminus (X_0 \cup X_1)$

Pour répondre aux deux types de questions suivantes :

a) Est-il inutile de résoudre ce problème (P) ?

en d'autres termes,

est-il possible de prouver que

$\nexists x \in \Gamma(P) : cx > \underline{v}(B)$?

où $\underline{v}(B)$ est un minorant de v(B) déterminé au préalable par une heuristique (c'est la notion *d'évaluation* par excès qui est mise alors en jeu).

S'il est répondu positivement à cette question, il est inutile d'explorer la partie de l'arborescence associée à X_2 (il faut alors considér une autre partition de $\{1,\ldots,n\}$).

S'il est répondu négativement, l'autre question est la suivante :

b) Est-il possible de réduire la taille de (P) ?
en d'autres termes
Est-il possible de mettre en évidence un sous-ensemble X_3 de X_2 tel que la résolution de
$(P \mid x_j = \varepsilon_j)\ j \in X_3,\ \varepsilon_j \in \{0,1\}$
est inutile ? (voir a))

Si la réponse est positive, la réduction de la taille de (P) obtenue en posant $x_j = 1 - \varepsilon_j\ \forall j \in X_3$ (le terme "élimination" des variables d'indice $j \in X_3$ sera également employé) aboutit au nouveau sous-problème

$$(P \mid x_j = 1 - \varepsilon_j \quad \forall j \in X_3).$$

Les résultats suivants permettent de répondre à ces préoccupatio

Théorème 4

Si $\exists\ \bar{v}(P) : \bar{v}(P) \leq \underline{v}(B)$ (5)
alors $\forall x \in F(P) : cx \leq \underline{v}(B)$

Démonstration

Par définition $v(P) \leq \bar{v}(P)$ (5) $\Big\} \Rightarrow v(P) \leq \underline{v}(B).$

□

Corollaire

Si $\exists\ \bar{v}(B) : \bar{v}(B) \leq \underline{v}(B)$
alors $v(B) = \underline{v}(B)$

Démonstration

Théorème 4 avec l'hypothèse supplémentaire $X_0 = X_1 = \emptyset$.

□

Théorème 5

Etant donnés $\varepsilon \in \{0,1\}$, $j \in X_2$
Si $\exists\, \bar{v}(P \mid x_j = \varepsilon) : \bar{v}(P \mid x_j = \varepsilon) \leq \underline{v}(B)$
alors $x_j \leftarrow 1 - \varepsilon$ *sous l'hypothèse* $x_k = 1\ \forall k \in X_1$ et $x_k = 0\ \forall k \in X_0$

Démonstration

$$\text{Si } \exists\, \varepsilon \in \{0,1\} : \bar{v}(P \mid x_j = \varepsilon) \leq \underline{v}(B) \left.\begin{matrix} \\ \\ \text{théorème 4} \end{matrix}\right\} \Rightarrow \forall x \in F(P \mid x_j = \varepsilon)\ cx \leq \underline{v}(B)..$$

$$\left.\begin{matrix} \ldots \Longrightarrow \text{ il est inutile de résoudre } (P \mid x_j = \varepsilon) \\ v(P) = \max\{v(P \mid x_j = 0),\ v(P \mid x_j = 1)\} \end{matrix}\right\} \Longrightarrow \text{ il faut}$$

considérer le sous-problème $(P \mid x_j = 1 - \varepsilon)$, d'où la conclusion.

Corollaire

□

Etant donnés $\varepsilon \in \{0,1\}$, $j \in \{1,\ldots,n\}$
Si $\exists\, \bar{v}(B \mid x_j = \varepsilon) : \bar{v}(B \mid x_j = \varepsilon) \leq \underline{v}(B)$ *alors* $x_j \leftarrow 1 - \varepsilon$

Démonstration

Théorème 5 avec l'hypothèse supplémentaire $X_0 = X_1 = \emptyset$.

□

Notes

(i) En notant $\underline{x} \in F(B)$ la solution associée à $\underline{v}(B)$

$$\underline{\text{si}}\ \exists\ \varepsilon \in \{0,1\},\ j \in \{1,\ldots,n\} : \begin{cases} v(B \mid x_j = \varepsilon) = \bar{v}(B \mid x_j = \varepsilon) = \\ v(B \mid x_j = 1 - \varepsilon) < \underline{v}(B) < \\ \qquad\qquad\qquad\qquad \bar{v}(B \mid x_j = 1 \end{cases}$$

le corollaire du théorème 5 nécessite la considération du sous-problème $(B \mid x_j = 1 - \varepsilon)$ bien que $v(B) = \underline{v}(B)$ et $\exists x^*$: $x^*_j = 1 - \varepsilon$

(ii) $j \in \{1,\ldots,n\}$ et $\varepsilon \in \{0,1\}$ étant donnés, un majorant $\bar{v}(P \mid x_j = \varepsilon)$ de $v(P \mid x_j = \varepsilon)$ est fourni par la valeur d'une relaxation de $(P \mid x_j = \varepsilon)$. Les sommets du graphe de dominance suivant donne un échantillon de relaxations qui seront utilisées par la suite :

i.e. $k \in I \subset \{1,\ldots,m\}$

u vecteur ligne de dimension convenable

étant donnés

(P_k) max cx s.c. $A_k x \leq b_k$; $x \in V$

(P_I) max cx s.c. $A_I x \leq b_I$; $x \in V$

$(RP_k(u))$ $\max_x cx + u(b_k - A_k x)$ s.c. $x \in V$

$(RP_I(u))$ $\max_x cx + u(b_I - A_I x)$ s.c. $x \in V$

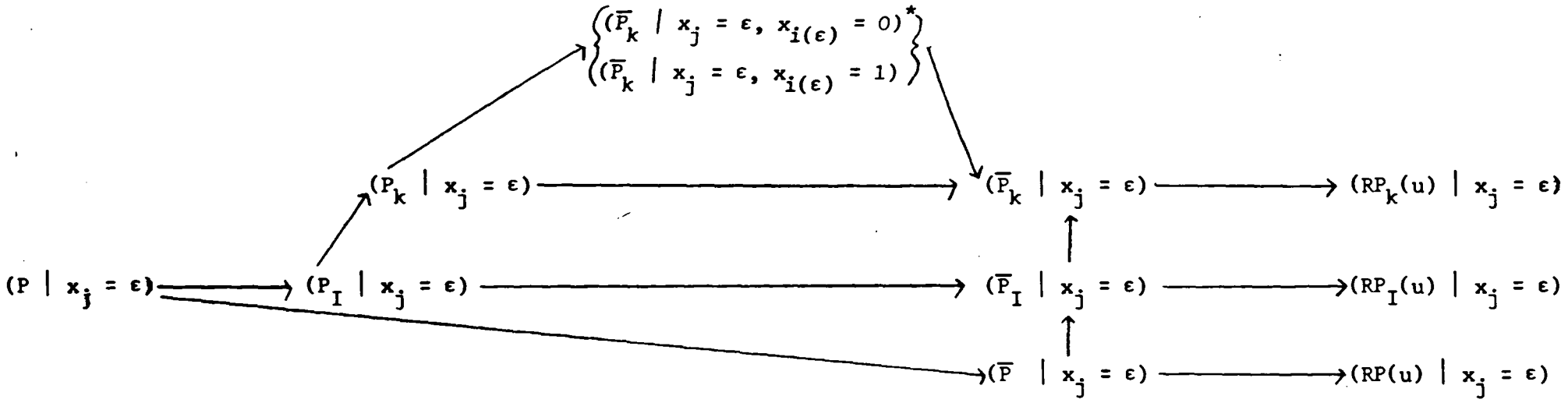
(et les relaxations et séparations associées)

les arcs du graphe sont tels que

$$(P_1) \rightarrow (P_2) \equiv v(P_1) \leq v(P_2)$$

(ce qui est bien entendu équivalent à : "(P_1) plus complexe à résoudre que (P_2)").

□



* $x_{i(\varepsilon)}$ est variable de base de $(\bar{P}_k \mid x_j = \varepsilon)$

Les résultats précédents conduisent à l'énoncé d'un algorithme général de réduction de la taille d'un problème de maximisation (B) qui sera utilisé de manière plus sophistiquée dans la plupart des méthodes exposées ci-après :

Algorithme de réduction

1 Résolution de $(\bar{B})$
: si $\bar{x} \in V$ alors $v(B) \leftarrow v(\bar{B})$; stop

2 Utiliser une (ou plusieurs) heuristique (s) pour déterminer $\underline{v}(B)$.

3 Déterminer un majorant de v(B) (par exemple $\bar{v}(B) = \lfloor v(\bar{B}) \rfloor$)
si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$; stop
$X_0 \equiv X_1 \equiv \emptyset$

4 Réduction du nombre de variables de (B) :
$\forall j \in \{1,\ldots,n\}$
si $\exists\, \varepsilon \in \{0,1\} : \bar{v}(B \mid x_j = \varepsilon) \leq \underline{v}(B)$ alors $x_j \leftarrow 1 - \varepsilon$
$X_{1-\varepsilon} \equiv X_{1-\varepsilon} \cup \{j\}$

sinon si $\underline{v}(B \mid x_j = \varepsilon) > \underline{v}(B)$ alors
$\underline{v}(B) \leftarrow \underline{v}(B \mid x_j = \varepsilon)$
si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$

Notes

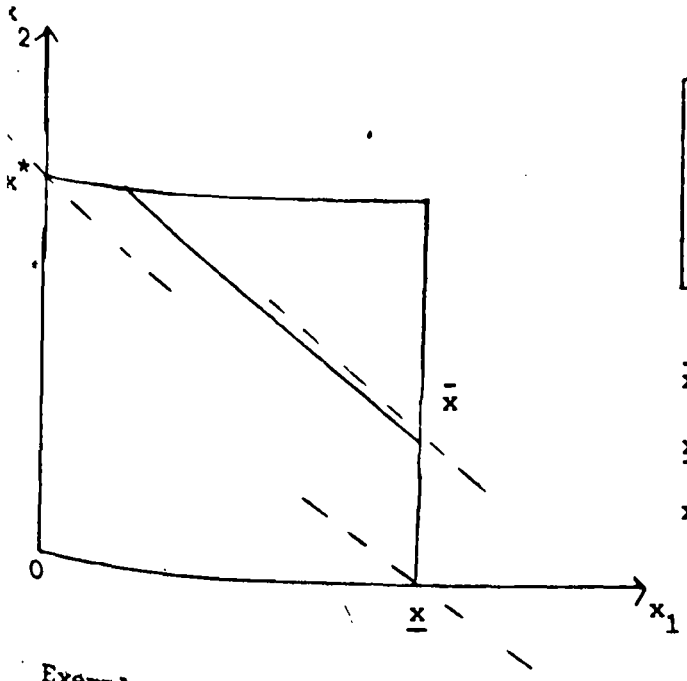
(i) il est clair que toutes ces considérations sont applicables aux problèmes de minimisation à condition

- d'échanger les rôles de $\underline{v}(.)$ et $\bar{v}(.)$, et de $\lfloor . \rfloor$ et $\lceil . \rceil$
- d'inverser les sens des inégalités.

(ii) dans la majorité des cas (sauf lorsque A est totalement unimodulaire), la résolution de $(\bar{B})$ ne peut apporter des informations suffisantes sur la solution optimale de (B) ; les deux exemples suivants donnent une illustration de cette assertion :

Exemple 1

Considérer la solution entière réalisable la plus "proche" de la solution optimale de $(\bar{B})$ peut aboutir au point entier diagonalement opposé à la solution optimale de (B) :



$$\begin{cases} \text{Max } 5x_1 + 6x_2 \\ \text{s.c. } 4x_1 + 5x_2 \leq 6 \\ \quad x_1, x_2 = 0 \text{ ou } 1 \end{cases}$$

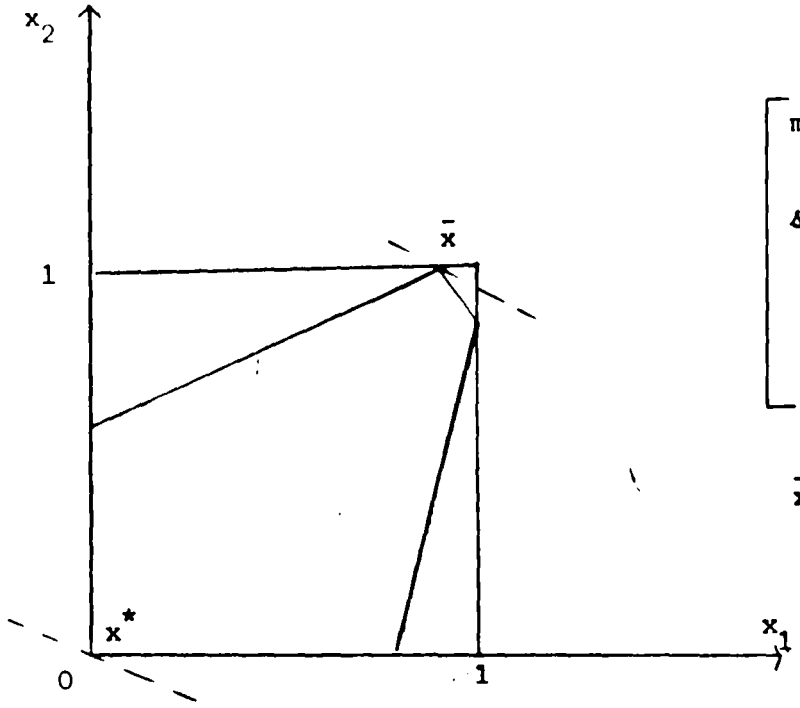
$\bar{x} = \begin{pmatrix} 1 \\ .4 \end{pmatrix}$, $v(\bar{B}) = 7.4$

$\underline{x} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $c\underline{x} = 5$

$x^* = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, $v(B) = 6$

Exemple 2

La solution de $(\bar{B})$ peut être diagonalement opposée à la solution de (B) (la solution $\bar{x}$ "arrondie" n'est pas réalisable pour (B)) :



$$\left[\begin{array}{l} \max \; x_1 + 2\,x_2 \\ \\ \text{s.c.} \; -20x_1 + 45x_2 \leq 27 \\ \qquad\; 10x_1 + 10x_2 \leq 19 \\ \qquad\; 45x_1 - 10x_2 \leq 36 \\ \qquad\; x_1,\, x_2 = 0 \text{ ou } 1 \end{array}\right.$$

$\bar{x} = \begin{pmatrix} .9 \\ 1 \end{pmatrix}$ et $x^* = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

□

II - COMPLEXITE DES ALGORITHMES

Dans ce domaine de la programmation mathématique pour lequel les variables sont astreintes à prendre des valeurs discrètes, les problèmes de convergence des méthodes de résolution ne se posent pas : l'ensemble des solutions de tout problème étant fini, il suffit, à la limite, de construire un algorithme explorant exhaustivement cet ensemble pour obtenir la ou une solution optimale en un *nombre fini d'étapes*, donc en un nombre fini d'opérations.

Fournir des informations sur le coût d'un algorithme c'est-à-dire sur le temps d'exécution et l'encombrement mémoire nécessaires à son implantation sur ordinateur, c'est l'objet de l'analyse de la complexité des algorithmes qui peut se scinder en deux classes :

(i) *la complexité dite "expérimentale"* qui indique une mesure précise du coût de l'algorithme, c'est-à-dire une mesure fonction des choix : d'utilisation d'une machine bien définie, d'un langage, d'un compilateur... Par exemple, dans cette optique d'une analyse qui peut être qualifiée d'"a posteriori", il peut exister un temps d'exécution propre à chaque type d'opérations.

(ii) *la complexité dite "théorique"* qui donne un ordre de grandeur du coût de l'algorithme, indépendamment du type de machine utilisée ; les temps d'exécution et l'espace mémoire de l'algorithme sont en général des fonctions - notées respectivement T et E - de la taille du problème (i.e. du nombre n de variables et du nombre m de contraintes).

C'est ce deuxième type d'analyse - pouvant être qualifié d'"a priori" - qui fait l'objet de notre étude.

Quelques notions générales sur la complexité théorique des algorithmes (§ II.1) précédent un résumé sur la théorie des problèmes réputés difficiles - appelés *NP-complets* - (§ II.2) débouchant sur les définitions relatives aux méthodes dites "approchantes" (§ II.3) qui seront étudiées en détail au chapitre 2 (§ III.1.2 et III.1.3.).

II.1 - NOTIONS GÉNÉRALES

Définition 4

Si pour un algorithme donné,

$\exists k, h \in \mathbb{R}$, f et g : $\mathbb{N}^2 \to \mathbb{N}$

$\exists n_0, m_0 \in \mathbb{N} : \forall n \geq n_0 \; \forall m \geq m_0 \begin{cases} T(n,m) \leq k\, f(n,m) \\ E(n,m) \leq h\, g(n,m) \end{cases}$

alors cet algorithme est dit de complexité $O(f(n,m))$ *en temps de calcul et* $O(g(n,m))$ *en encombrement mémoire.*

□

Notes

(i) Des paramètres autres que ceux caractérisant la taille du problème peuvent être introduits ; deux exemples seront fournis aux chapitres 2 (§III.1) et 4 (§ IV.4).

(ii) Les paramètres n et m qui caractérisent la taille du problème doivent être en fait une mesure de la "longueur" de la représentation des données du problème.

Par exemple, on peut imaginer que deux exemplaires d'un même problème donné nécessitent deux types de représentation "machine" différentes compte-tenu des valeurs respectives de leurs données :

Si dans la représentation la plus simple c'est-à-dire une machine dite "par mot", le premier nécessite un mot par donnée et le deuxième deux mots par donnée, un algorithme dit de complexité $O(2^n)$ dans la terminologie précédente, aboutit en fait à un temps de calcul proportionnel à 2^n pour le premier exemplaire et à 2^{2n} pour le deuxième exemplaire.

Bien que de nombreux articles sur la complexité évoquent la longueur d'entrée des données en représentation binaire [C12, C20, C23], dans tout ce qui suit on supposera la mise en oeuvre des algorithmes sur une machine à mot, que tous les exemplaires d'un même problème auront la même représentation (nombres de mots par donnée identiques), et enfin que toutes les opérations seront équivalentes du point de vue coût.

(iii) Tous les raisonnements qui suivront seront faits pour la complexité en temps de calcul (on supposera toujours savoir implanter les algorithmes en mémoire).

□

Dans toute la suite, on supposera la taille du problème uniquement définie par le nombre n de variables [c'est le cas lorsque m = 1] ; les types de complexité rencontrés le plus souvent seront $O(n^k)$ (avec $k \in \mathbb{R}^+$) [la complexité sera alors dite *polynômiale*], O(nlogn) et $O(2^n)$.

L'objection la plus évidente à formuler sur cette définition de la complexité est la suivante : seul le comportement asymptotique de l'algorithme est envisagé puisque les constantes multiplicatives sont négligées ; ainsi un algorithme dont le temps de calcul est une fonction du type $n^2 + 2^{20}n + 2^{30}$ sera dit de complexité $O(n^2)$ bien que $2^{30} > 2^{20}$ n pour $n < 2^{10}$ et $2^{20}n > n^2$ pour $2^{10} \leq n < 2^{20}$... Dans un même ordre d'idée, il faut remarquer que $2^n < 103n$ $\forall n \leq 10$...

Sans perdre de vue cette critique [pour les problèmes de petite taille, il sera intéressant de remettre en jeu les constantes de proportionnalité ; par exemple dans le domaine du classement des éléments d'un fichier, le tri dit par "insertion" est plus performant que le tri appelé "quicksort" pour n ≤ 10] deux arguments significatifs viennent étayer la notion de complexité de la définition 4 :

Les progrès de la technologie font que :

- d'une part, l'accroissement des tailles maximales des problèmes acceptées par les ordinateurs donne de moins en moins d'importance aux constantes multiplicatives.

- d'autre part, la recherche d'algorithmes performants du point de vue de la complexité théorique de la définition 4 deviendra de plus en plus cruciale ; l'exemple suivant, théorique mais significatif, illustre cette dernière assertion :

En supposant que :

- pendant un temps donné t, une machine M_1 puisse réaliser 2^{10} opérations

- le traitement d'une donnée parmi les n d'un problème (P) nécessite une opération unique.

le tableau suivant indique pour quatre algorithmes de complexités différentes, les tailles maximales de (P) pouvant être traitées pendant le temps t : d'une part sur la machine M_1 et d'autre part sur une machine M_2 supposée seize fois plus rapide.

Algorithme	Complexité	Tailles maximales		
		M_1	M_2	
1	$O(2^n)$	10	14	(10+4)
2	$O(n^2)$	32	128	(32x4)
3	$O(n)$	1024	16384	(1024x16)
4	$O(\log_2 n)$	2^{1024}	$2^{16384}((2^{1024})^{16})$	

Tableau 1

II.2 - PROBLÈMES NP-COMPLETS

Après avoir noté que l'analyse de la complexité d'un algorithme doit être faite indépendamment des données du problème traité, deux types d'analyse sont rencontrés dans la littérature :

- l'analyse du *cas le plus défavorable* - notée D-analyse - qui considère la complexité de l'algorithme pour les n données du problème fournissant le temps d'exécution le plus long.

- l'analyse *probabiliste* (ou *moyenne*) - notée P-analyse - qui au contraire de la D-analyse, cherche à borner le temps d'exécution dans la "plupart des cas" ; i.e. connaissant la loi de répartition des données du problème, elle détermine la probabilité pour que le temps d'exécution soit inférieur à un seuil donné ([C5, C13, C18]).

Il est clair que la D-analyse et la P-analyse offrent des avantages et des inconvénients différents pour atteindre l'objectif de base qui est la prévision des performances d'un algorithme ; c'est pourquoi ces analyses associées à l'analyse empirique (expériences sur ordinateur) doivent être considérées comme complémentaires et non en opposition.

Bien que la D-analyse soit a priori la moins proche de l'analyse expérimentale, elle offre l'avantage de ne faire aucune hypothèse sur les données du problème et présente donc un caractère plus général que la P-analyse.

C'est dans ce cadre de la D-analyse qu'a été développée l'étude d'une grande classe de problèmes combinatoires probablement non traitables numériquement par quelque algorithme que ce soit.

Plus précisément la conjecture est la suivante : *tout algorithme pour l'un des problèmes de la classe* doit, dans le cas le plus défavorable, nécessiter un *temps d'exécution croissant exponentiellement avec la taille du problème* ; deux types d'étude viennent étayer cette assertion :

- *Etude expérimentale* : tous les algorithmes connus à ce jour pour ces problèmes sont de complexité exponentielle.

- *Etude théorique* : Cook [C4] et Karp [C11] ont défini la classe des problèmes dits NP-complets qui sont équivalents dans le sens suivant : si l'un des problèmes de cette classe pouvait être résolu en un temps polynômial alors ce serait le cas pour tous les autres problèmes de la classe.

La démarche de l'utilisation de ces problèmes dans le cadre de l'optimisation est maintenant résumée avant de définir plus précisément la terminologie employée :

Un problème d'optimisation (P) étant donné, on suppose qu'il n'existe pas en l'état actuel de la connaissance d'algorithme de complexité polynômiale pour le résoudre sur ordinateur (dans le cas contraire ce problème est considéré comme "facile"). Dans le but de déterminer s'il y a une grande probabilité pour qu'il n'existe jamais un tel algorithme

((P) appartiendrait à une classe de problèmes que nous appelerons *Ardus*) on considère le *problème de décision* associé (DE) (i.e. problème d'existenc de solution), plus facile à résoudre à priori (le résultaq fourni est vrai ou faut).

Si (DE) appartient à la classe dite *P* des problèmes les plus facil alors on ne peut conclure que (P) est *Ardu*. Si, par contre, (DE) appartient à la sous-classe appelée *NP*-complet des problèmes dits *NP* alors (P) est dit *Ardu*.

II.2.1 - Algorithmes non déterministes et problèmes de décision

Tous les algorithmes évoqués jusqu'à présent sont des *algorithmes déterministes* : ce sont des algorithmes classiques destinés à la mise en oeuvre sur ordinateur, pour lesquels le *résultat* de chaque opération est défini de *manière unique*.

D'un point de vue théorique, il est alors supposé *l'existence*

- d'une part d'algorithmes dits *non déterministes* pour lesquels chaque opération peut admettre tout un éventail S de résultats possibles.

- d'autre part d'une machine "idéale" dite *non déterministe* qui, en cours d'exécution de tels algorithmes, peut choisir un résultat quelcon de S (en un temps d'exécution $O(1) \forall |S|$).

Plus précisément un algorithme non déterministe n'admet que le ré sultat unique : *Succès* ou *Echec* suivant que le problème traité a *ou* n'a p de solution ; il comporte en outre une fonction appelée *choix* de S dans S dont la valeur est fixée par la machine non déterministe qui est ainsi sup sée fournir pour chaque opération de l'algorithme le *ou* les *bon(s) choix* a tissant à la réponse positive *Succès* de l'algorithme, si elle existe. Ain la réponse d'un algorithme non déterministe sera *Echec* si et seulement si n'existe aucune voie (i.e. aucun choix) pour aboutir à une réponse positi

Exemple : Un algorithme non déterministe recherchant l'existence d'une valeur x dans un fichier $a_1, a_2, \ldots, a_n$ est le suivant :

0 x et $a_1, a_2, \ldots, a_n$ donnés
1 j ← *choix* (1,2,...,n)
2 Si $a_j = x$ alors imprimer j ; *Succès*
sinon *Echec*

Un tel algorithme est de complexité O(1) ; en effet la machine non déterministe est supposée trouver directement par l'intermédiaire de la fonction *choix* l'existence ou la non existence d'un indice j de {1,...,n} tel que $a_j = x$. Il est à noter que l'algorithme déterministe réalisant ce traitement serait de complexité O(n).

□

Pour de nombreux *problèmes de décision* ne pouvant être résolus en temps polynômial par un algorithme déterministe, il existera par contre un algorithme non déterministe de complexité polynômiale ; c'est le cas pour le problème de décision du knapsack à n variables bivalentes qui peut se formuler comme suit :

(DE) Etant donnés | (B) max cx s.c. ax ≤ b ; x ∈ V
et $\underline{v}$(B) un minorant de v(B)
Existe-t-il x ∈ V : cx > $\underline{v}$(B) et ax ≤ b ?

L'algorithme non déterministe suivant de résolution de (DE) est à l'évidence de complexité O(n) :

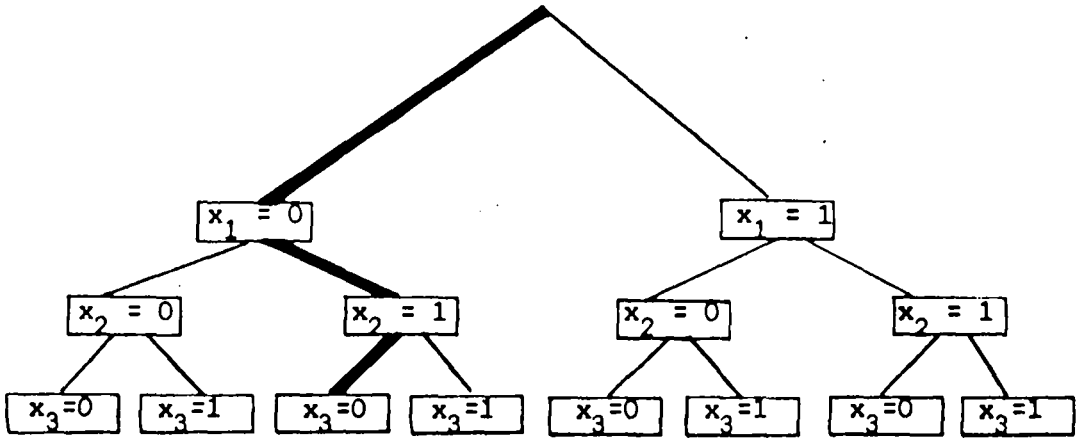
0 n, $a_1, \ldots, a_n$, $c_1, \ldots, c_n$, b donnés
1 x_j ← *choix* (0,1) j = 1,...,n
2 si cx > $\underline{v}$(B) et ax ≤ b alors *Succès*
sinon *Echec*

Bien que (DE) soit plus simple dans sa formulation, il est aussi complexe à résoudre que le problème (B) : il n'existe pas d'algorithme déterministe polynômial donnant une solution de (DE) puisqu'a priori (toujours dans le contexte du cas le plus défavorable) il faudra explorer les 2^n sommets du cube unité. Ce n'est pas le cas pour l'algorithme non déterministe qui, à chaque étape de l'exploration de l'arborescence, réalisera le *bon choix* d'affectation de la valeur 0 ou 1 à chacune des n variables :

Exemple

Etant donnés

$$\begin{cases} (B) \quad \max 3x_1 + 4x_2 + x_3 \text{ s.c. } 2x_1 + 3x_2 + 5x_3 \leq b \; ; \; x \in V \\ \text{et} \\ (DE) \; \exists x \in V : 3x_1 + 4x_2 + x_3 > \underline{v}(B) \text{ et } 2x_1 + 3x_2 + 5x_3 \leq b \; ? \end{cases}$$



cx	0	1	4	5	3	4	7	8
ax	0	5	3	8	2	7	5	10

L'arborescence ci-dessus couplée au tableau des 2^3 valeurs possibles du couple (cx, ax), $x \in V$, fournit une simulation déterministe du déroulement de l'algorithme non déterministe :

- en cas de réponse *Succès*, une seule branche de l'arbre est parcourue (i.e. celle en trait gras, lorsque $\underline{v}(B) = 3$ et $b = 4$)

- en cas de réponse *Echec*, les huit branches de l'arbre sont explorées *en un temps* O(1) : ce qui correspondrait à l'existence d'une machine "surdouée" qui pourrait effectuer autant de calculs en parallèle qu'il est nécessaire !

II.2.2 - Problèmes *Ardus*

Définition 5 : [C4, C12, C20]

La classe P est celle des problèmes de décision pour lesquels il existe un algorithme déterministe de complexité polynômiale.

Définition 6 : [C4, C12, C20]

La classe NP est celle des problèmes de décision pour lesquels il existe un algorithme non déterministe de complexité polynômiale.

A l'évidence P est un sous-ensemble de NP. Est-il inclus strictement dans NP ? Cette question n'a connu, à l'heure actuelle, aucune réponse, bien que la conjecture soit $P \neq NP$.

La théorie venant étayer cette conjecture a pour base la définition d'une sous-classe de NP, de problèmes a priori plus difficiles que les autres :

Définition 7 [C4, C12, C20]

Les problèmes NP-complets forment la sous-classe de NP définie comme suit :

$\forall$(P) $\in$ NP-complet, s'il existait un algorithme déterministe de complexité polynômiale pour (P) alors il en existerait un pour tout problème de NP.

Ainsi, s'il était possible de résoudre par un algorithme déterministe de complexité polynômiale un quelconque des problèmes *NP*-complets alors *P = NP*.

Définition 8

Un problème d'optimisation sera dit Ardu si le problème de décision associé est NP-complet.

Donc un problème d'optimisation *Ardu* ne peut (en l'état actuel de nos connaissances) être résolu théoriquement par un algorithme de complexité polynômiale.

On peut introduire, à l'instar de Garey, Johnson [C8], la notion de problèmes *Fortement Ardus* pour lesquels il n'existe même pas d'algorithme dit *pseudo-polynômial*, c'est-à-dire un algorithme dont le temps de calcul est borné par un polynôme des deux variables : "taille du problème" et "un paramètre spécifique par exemplaire considéré".

Exemple : On rappelle au chapitre 2 § III.1.3 que la résolution par la programmation dynamique du problème du knapsack est un algorithme de complexité $O(nb)$, où b est la valeur du second membre du problème considéré.

Le problème du knapsack qui entre dans la catégorie des problèmes *Ardus* n'est donc pas *Fortement Ardu* ; il est parmi les moins difficiles à résoudre en théorie.

Notes

(i) La théorie des problèmes *P*, *NP* et *NP*-complets est issue de l'étude théorique des langages [*NP* étant un sous-ensemble des langages récursifs qui eux-mêmes sont inclus dans l'ensemble des languages récursivement énumérables] ; il est donc supposé savoir associer à tout problème de décision un langage de la même classe c'est-à-dire *P* ou *NP*.

(ii) Pour une bibliographie détaillée voir Garey, Johnson [C7] Sahni et Horowitz [C20] et Weide [C23].

□

II.2.3 - Méthodes approchantes

En se plaçant du point de vue économique, des problèmes dits *Ardus* sont à résoudre chaque jour dans les entreprises, mais le terme "résolution" n'a alors pas le sens théorique classique : seule une bonne solution réalisable est demandée ; sa détermination est souvent obtenue très simplement par l'emploi *d'heuristiques* (i.e. algorithmes pour lesquels la preuve d'optimalité n'est pas faite) qui non seulement "ne coûtent pas cher" mais de plus aboutissent à des valeurs proches de l'optimalité.

Comme dans le cas de la complexité des algorithmes, deux types d'analyses théoriques des heuristiques sont proposées dans la littérature :

- l'analyse du *cas le plus défavorable* - appelée D-analyse - qui considère le (ou les) exemplaire (s) du problème traité fournissant l'écart maximum entre la valeur optimale et celle donnée par l'heuristique.

- l'analyse *probabiliste* - appelée P-analyse - ; une hypothèse sur la loi de répartition des données du problème étant faite, elle établit des propriétés probabilistes de l'heuristique telles que : la détermination d'une borne sur la probabilité de trouver un écart entre la valeur optimale et celle de l'heuristique, inférieur à un seuil donné.

La terminologie employée pour l'étude de la D-analyse des heuristiques a été introduite par Garey, Johnson [C7] (voir également [A3, A8, C19]), elle sera utilisée pour l'analyse détaillée des heuristiques pour le problème du knapsack (voir chapitre 2 § III.1).

Etant donnés un problème de maximisation (P), considéré en tant que collection de tous ses exemplaires numériques notés (E), et un algorithme fournissant un minorant $\underline{v}(E)$ de $v(E)$ (supposé posifif) quelque soit l'exemplaire numérique (E) de (P) :

Définition 9

L'algorithme est dit ε-approchant pour le problème (P) si
$\exists\ \varepsilon \in [0,1[$:
quelque soit l'exemplaire numérique (E) du problème (P)

$$(v(E) - \underline{v}(E))/v(E) \leq \varepsilon \qquad (6)$$

Notes

(i) La relation (6) est équivalente à

$$\underline{v}(E) \geq (1 - \varepsilon)\, v(E) \qquad (6')$$

(ii) Les relations (6) et (6') indiquent implicitement que

$$\exists (E) : \underline{v}(E) = (1 - \varepsilon)\, v(E)$$

ou

$$\exists \text{ suite infinie } (E_k)_{k \in \mathbb{N}} \text{ d'exemplaires de } (P) :$$

$$\underline{v}(E_k)/v(E_k) \xrightarrow[k \to +\infty]{} 1 - \varepsilon$$

(iii) Il est clair que la définition n'a pas de sens pour $\varepsilon \notin [0,1[$.

□

Définition 10

Un algorithme approchant pour (P) est un algorithme vérifiant (6) $\forall\, \varepsilon \in]0,1[$.

Note

Un algorithme approchant est donc "ajustable" suivant le degré de précision $\varepsilon \in]0,1[$ requis ; ce n'est pas le cas pour un algorithme ε-approchant qui ne peut garantir au mieux qu'une valeur approchée à ε près, ε *étant imposé*.

Définition 11

Un algorithme approchant polynômial pour (P) est un algorithme approchant dont le temps d'exécution est borné par une fonction polynômiale de la taille du problème.

Note

Un tel algorithme sera appelé P.A. (pour Polynomial time Approximate).

□

Définition 12

Un algorithme approchant totalement polynômial est un algorithme approchant dont le temps d'exécution est borné pour une fonction polynômiale à la fois de la taille du problème et de l'inverse de la précision.

Note

Un tel algorithme sera appelé F.P.A. (pour Fully Polynomial time Approximate).

□

LE PROBLEME DU KNAPSACK

EN VARIABLES 0-1

CHAPITRE 2

INTRODUCTION

L'intérêt indéniable du problème du knapsack en variables bivalentes

$$(B)\quad \begin{cases} cx^* = \max\ cx \\ \quad s.c.\ ax \leq b \\ \quad\quad x \in V \end{cases}$$

$$\text{avec}\quad \begin{cases} c,\ a \in \mathbb{N}^{*n} \text{ et } b \in \mathbb{N}^* \\ \max\limits_{1\leq j\leq n} a_j \leq b < \sum\limits_{j=1}^{n} a_j \text{ (voir note)} \end{cases}$$

dans le cadre de la résolution de problèmes concrets de grande taille ([G11, G28, G37, G39, G42], partie I et chapitre 3) a conduit de nombreux auteurs à élaborer des méthodes de résolution de (B) appartenant à deux grandes classes :

(i) *les méthodes approchantes* : Le problème de décision du knapsack étant *NP*-complet (voir chapitre 1, partie II) certains auteurs [A1 à A12] ont construits des algorithmes de complexité polynômiale c'est-à-dire qui fournissent une valeur approchée de v(B) avec une précision donnée a priori, en un temps polynômial fonction de la taille de (B) et de l'inverse de la précision, pour les plus performantes d'entre elles (§ III.1.3).

(ii) *les méthodes exactes* : les cas les plus défavorables exhibés en théorie n'apparaissant pas en général dans la pratique - un parallèle est à faire avec la méthode simpliciale qui n'est pas de complexité polynomiale en théorie [C14] - une majorité d'auteurs ont mis en oeuvre des algorithmes énumératifs aboutissant à la solution exacte de (B) en des temps de calcul empiriques très acceptables.

Il est actuellement clair que les plus efficaces d'entre eux (Balas, Zemel [K2], Barr, Ross [K3], Fayard, Plateau [K8 à K16], Guignard, Spielberg [K24], Hansen [K26], Ingargiola, Korsh [K28], Laurière [K31], Martello, Toth [K34], Nauss [K37], Suhl [K41], Walukiewicz [K44], Zoltners [K45]) comportent trois phases successives :

Phase 1 : résolution de la relaxation de (B) ;
$v(\bar{B}) = \max \{cx \mid ax \leq b,\ x \in [V]\}$

Phase 2 : réduction de la taille de (B) (élimination définitive de variables après attribution de valeurs bien définies)

Phase 3 : résolution du problème réduit par une méthode d'énumération implicite.

Cette dernière phase n'occupant qu'une place infime, du point de vue temps de calculs, par rapport aux phases 1 et 2 - par exemple, pour les problèmes de 5 000 variables, les temps requis pour la phase 3 sont de l'ordre de 7.4 % des temps globaux - nous nous sommes uniquement attachés à rappeler (partie VI) deux principes d'énumération :

- la procédure dite LIFO (Last in First Out) prônée par Balas [G1, G2] et Geoffrion [G20] et utilisée dans la majorité des méthodes pour le knapsack

- la procédure dite lexicographique, exposée dans [K8, K9, K10]

Par contre, notre intérêt s'est porté plus particulièrement sur l'étude approfondie des deux premières phases pour lesquelles nous proposons de faire le point en apportant à chaque fois notre contribution.

Phase 1 (partie II) : la méthode classique de résolution de $(\bar{B})$ comporte le tri initial des données suivant l'ordre décroissant des c_j/a_j $j = 1,\cdots$ La part énorme des temps de calcul consacrés à ce tri par rapport aux temps globaux de résolution [K12] a conduit à la mise au point d'une nouvelle méthode très efficace ne nécessitant plus le tri systématique des données à titre d'exemple, cette méthode de complexité $0(n)$ aboutit à un gain de l'ordre de 74 % pour les problèmes tirés au hasard de 9 000 variables.

Phase 2 : l'exposé d'un catalogue de méthodes d'encadrement sur la fonction économique et sur la contrainte (i.e. déterminant $\underline{v}(B)$ (§III.1), $\bar{v}(B)$ (§ I
et $\underline{b}$ (§ III.3) telles que (B) peut être remplacé par

max cx s.c. $\underline{v}(B) < cx \leq \bar{v}(B)$; $\underline{b} \leq ax \leq b$; $x \in V$)

et de résultats théoriques relatifs aux relaxations lagrangiennes de (B) (partie IV) précède la présentation d'un cadre commun pour les algorithmes de réduction de la taille de (B) et d'une nouvelle méthode performante mettant en jeu des relaxations, des relaxations lagrangiennes, des relations binaires et de dominance entre variables (partie V).

Les expériences numériques - réalisées sur UNIVAC 1110 - relatives à ces trois phases sont résumées en partie VII.

Enfin, il est à remarquer que la résolution de (B) par la programmation dynamique est évoquée au § III.1 en résultats préliminaires à l'exposé des algorithmes approchants totalement polynômiaux.

Note

L'hypothèse

$$\begin{cases} c_j,\ a_j > 0 & \forall j \in \{1,\ldots,n\} \\ \max\limits_{1\leq j\leq n} a_j \leq b < \sum\limits_{j=1}^{n} a_j \end{cases}$$

permettant d'éliminer les solutions triviales, découle des remarques suivantes :

si, a priori , les données b, c_j, $a_j (j = 1,\ldots,n)$ sont de signes quelconques, alors :

(i) des changements de variables de la forme

$$x'_j = 1 - x_j$$

permettent d'obtenir des coefficients c_j tous non négatifs,

(ii) en posant alors

$$J^+ = \{j \mid a_j > 0\},\ J^- = \{j \mid a_j \leq 0\}$$

il est clair que $x^*_j = 1,\ \forall j \in J^-$

(iii) si $b + \sum_{j \in J^-} |a_j| < 0$ alors le domaine de (B) est vide,

(iv) si $b + \sum_{j \in J^-} |a_j| = 0$ alors $x_j^* = 0 \ \forall j \in J^+$

(v) si $\sum_{j \in J^+} a_j \leq b + \sum_{j \in J^-} |a_j|$ alors $x_j^* = 1 \ \forall j \in J^+$

(vi) lorsque (v) n'est pas vérifié,
<u>si</u> $\exists k \in J^+$ tel que $c_k = 0$ ou $a_k > b + \sum_{j \in J^-} |a_j|$
<u>alors</u> $x_k^* = 0$.

□

I - INTERET DU PROBLEME

L'intérêt de l'étude du problème du knapsack (en dehors de son interprétation économique propre) a de multiples facettes dans le cadre de la résolution de problèmes de grande taille ; quatre d'entre elles ont été retenues :

L'utilisation du problème du knapsack comme sous-problème permet

- soit d'obtenir un majorant du problème d'origine : l'exemple traité est celui du problème d'affectation généralisé (§ I.1).

- soit de rendre plus efficace une méthode d'optimisation : par exemple

. en accélérant la méthode simpliciale pour le problème de découpe à une dimension (§ I.2).

. en renforçant les coupes dans les méthodes algébriques (§ I.3).

La transformation d'un système d'équations diophantiennes en une équation unique équivalente (§ I.4) permet le traitement d'un problème à plusieurs contraintes par une méthode de résolution du problème du knapsack. Une illustration concrète sera fournie aux chapitres 3 (partie II) et 4 (partie IV).

I.1 - RELAXATIONS LAGRANGIENNES DU PROBLÈME D'AFFECTATION GÉNÉRALISÉ

L'exemple classique pour lequel la relaxation lagrangienne fournit des sous-problèmes du knapsack en variables 0 - 1 est le problème d'affectation généralisé [noté G.A.P.] (voir [G16, G42, G44, G51]).

Sa formulation est la suivante :

$$
(G.A.P.)\quad
\begin{cases}
\min \displaystyle\sum_{i\in I}\sum_{j\in J} c_{ij}\, x_{ij} & \\
\text{s.c.} \displaystyle\sum_{i\in I} x_{ij} = 1 & \forall j \in J \equiv \{1,\dots,n\} \quad (7)\\
\displaystyle\sum_{j\in J} a_{ij}\, x_{ij} \le b_i & \forall i \in I \equiv \{1,\dots,n\} \quad (8)\\
x_{ij} = 0 \text{ ou } 1 & \forall i \in I,\ \forall j \in J \quad (9)
\end{cases}
$$

L'interprétation économique du (G.A.P.) est l'affectation optimale de m machines à n tâches données.

- Connaissant les quantités toutes positives suivantes :

  c_{ij} = coût, a_{ij} = temps : nécessaire à la réalisation de la tâche j par la machine i, $\forall i$ $\forall j$

  b_i = capacité (en temps) de la machine i

- Sachant qu'une tâche ne peut être affectée qu'à une machine et une seule (contraintes (7)).

La variable d'état x_{ij} prend la valeur 1 (resp. 0) si la machine i est affectée à la tâche j (resp. sinon).

Note

Le fait qu'une machine peut réaliser plusieurs tâches (contrainte (8)) constitue la généralisation du problème d'affectation classique où contraintes du type (8) seraient $\sum_{j\in J} x_{ij} = 1 \quad \forall i \in I \Longrightarrow |I| = |J|$

[pour le (G.A.P.) (8) $\Longrightarrow \sum_{j\in J} (a_{ij}/b_i)\, x_{ij} \le 1$ et en général $|I| \neq |J|$].

□

Deux types de relaxations lagrangiennes génèrent des problèmes du knapsack en variables 0 - 1 :

Type 1 : la dualisation des containtes (8) aboutit au problème

$$(R.G.A.P.^1_{(u)}) \quad \left[\begin{array}{l} \min \sum_{i\in I} \sum_{j\in J} c_{ij}\, x_{ij} + \sum_{i\in I} u_i \left(b_i - \sum_{j\in J} a_{ij}\, x_{ij}\right) \\ \text{sous les contraintes (7) et (9) ; } u \in \mathbb{R}^n_- \end{array} \right.$$

dont la résolution est triviale puisque

$$v(R.G.A.P.^1_{(u)}) = \sum_{i\in I} u_i\, b_i + \min \Big\{ \sum_{j\in J} \sum_{i\in I} (c_{ij} - u_i\, a_{ij})\, x_{ij} \;\Big|\; \sum_{j\in I} x_{ij} = 1 \ \forall j;\ x_{ij} = 0 \text{ ou } 1 \ \forall i\ \forall j \Big\}$$

$$= \sum_{i\in I} u_i\, b_i + \sum_{j\in J} \min \Big\{ \sum_{i\in I} (c_{ij} - u_i\, a_{ij})\, x_{ij} \;\Big|\; \sum_{j\in I} x_{ij} = 1 \ ;\ x_{ij} = 0 \text{ ou } 1 \ \forall\, i \Big\}$$

$$= \sum_{i\in I} u_i\, b_i + \sum_{j\in J} \min_{i\in I} (c_{ij} - u_i\, a_{ij})$$

Notes

(i) la complexité de résolution de $(R.G.A.P.^1_{(u)})$ est $O(mn)$.

(ii) dans le cas particulier où $u = 0$, la valeur optimale correspond à l'affectation de chaque tâche à la machine de moindre coût sans se préoccuper des capacités des machines.

□

Type 2 : la dualisation des contraintes (7) :

$$(R.G.A.P.^2_{(u)}) \quad \left[\begin{array}{l} \min \sum_{i\in I} \sum_{j\in J} c_{ij}\, x_{ij} + \sum_{j\in J} u_j \left(1 - \sum_{i\in I} x_{ij}\right) \\ \text{sous les contraintes (8) et (9) ; } u \in \mathbb{R}^n \end{array} \right.$$

conduit à la résolution de m problèmes du knapsack non triviaux puisqu'en posant : $\forall i \in I$

$$(B^i(u)) \quad \min \sum_{j \in J} (c_{ij} - u_j)\, x_{ij} \quad \text{s.c.} \sum_{j \in J} a_{ij}\, x_{ij} \leq b_i \ ;$$

$$x_{ij} = 0 \text{ ou } 1 \quad \forall j \in J$$

on obtient

$$v(R.G.A.P^2(u)) = \sum_{j \in J} u_j + \sum_{i \in I} v(B^i(u))$$

I.2 - LE PROBLÈME DE DÉCOUPE À UNE DIMENSION

I.2.1 - Enoncé

Un fournisseur est supposé avoir en *stock* des pièces de métal rectangulaires de p types (p longueurs différentes $L_1, L_2, \ldots, L_p$, les largeurs étant toutes identiques) ; à chaque pièce d'un même type $k \in \{1,$ est attaché un coût c_k faisant intervenir entre autre la fabrication, le tr port, le stockage.

Les demandes de mêmes caractéristiques des différents clients sont supposés regroupées en *m postes de commandes* correspondants à d_i pièce de longueurs ℓ_i $(i = 1, 2, \ldots, m)$ qui sont (en général) à découper dans les pièces stockées.

En supposant que le coût de fabrication des $\sum_{i=1}^{m} d_i$ pièces est égal à la somme des coûts des pièces stockées utilisées (i.e. les chut résultant de la découpe d'une pièce stockée - pièces de métal non affectée un client - sont considérées comme *inutilisables*), le problème de découpe une dimension est donc la fabrication des m postes de commandes au moindre coût.

I.2.2 - Formulation du problème

La matrice A des contraintes étant construite de la manière suivante :

pour chaque type de pièce stockée - de longueur L_k, $k \in \{1,\ldots,p\}$ - sont envisagées toutes les possibilités de découpes satisfaisantes (i.e. aboutissant à une chute inférieure à un seuil M donné) pour générer comme suit $|J_k|$ colonnes de A :

$$\forall j \in J_k \quad 0 \leq L_k - \sum_{i=1}^{m} a_{ij}\, \ell_i \leq M$$

où a_{ij} = (nombre de découpes de longueur ℓ_i) $\in \mathbb{N}$ $\forall i$ $\forall j$

le problème de découpe se formule :

$$(CS) \quad \left[\begin{array}{l} \min \sum_{k=1}^{p} c_k \sum_{j \in J_k} x_j \\ \text{s.c.} \sum_{k=1}^{p} \sum_{j \in J_k} a_{ij}\, x_j = d_i \quad i = 1,\ldots,m \\ x_j \in \mathbb{N} \ \forall j \in \bigcup_{k=1}^{p} J_k \end{array} \right.$$

où x_j désigne le nombre d'utilisations du $j^{\text{ième}}$ type de découpe (représenté par la colonne A^j) $\forall j \in \bigcup_{k=1}^{p} J_k$. (on pose $n = \sum_{k=1}^{p} |J_k|$)

I.2.3 - Utilisation du problème du knapsack

La matrice A possédant un nombre important de colonnes - Gilmore cite dans [G24], des tailles courantes de problèmes concrets de l'ordre de ($m = 20$, $n = 24\,000$) pour les petites tailles et de ($m = 30$, $n = 190\,000$) pour les tailles "raisonnables" - les problèmes de mémorisation nécessitent la résolution de $(\overline{CS})$ en appliquant le schéma classique de la méthode simpliciale :

Le passage d'une base réalisable I à une base voisine est réalisé par la résolution de trois systèmes linéaires :

(i) trouver u : $u\,A^I = c_I$

(ii) si $\exists s \notin I : u\,A^s > c_s$ alors I est optimale ; stop.
soit $s \notin I : c_s < u\,A^s$

(iii) résoudre $A^I\,y = A^s$
si $y \leq 0$ alors $(\overline{CS})$ possède une valeur infinie, stop.

(iv) résoudre $A^I\,x = d$
pour déterminer r : $x_r/y_r = \min \{x_j/y_j \mid y_j > 0\}$

(v) poser $I \equiv I - \{r\} + \{s\}$; aller en (i)

La phase de recherche de $s \notin I : uA^s > c_s$ (et donc la détermination de l'optimalité) est accélérée par l'introduction de résolutions de problèmes du knapsack (p au maximum dans le cas de l'optimalité) suivants

Etant donné $k \in \{1,\ldots,p\}$:

$$(B^k) \quad \begin{cases} uz^k = \max u_1 z_1 + u_2 z_2 + \ldots + u_m z_m \\ \quad \text{s.c.} \quad \ell_1 z_1 + \ell_2 z_2 + \ldots + \ell_m z_m \leq L_k \\ \qquad z_j \in \mathbb{N} \quad j = 1,\ldots,m \end{cases}$$

La phase (ii) de la méthode simpliciale devient donc :

0 $k \leftarrow 1$
1 si $v(B^k) \leq c_k$ alors aller en 2
déterminer $s \in J_k : A^s \neq A^k$; aller en (iii)
2 si $k < p$ alors $k \leftarrow k + 1$; aller en 1
sinon I est optimale ; stop.

Note

Pour plus de détails, se référer à [K17, K18, G19, G24], et à [K19, G10] pour les problèmes de dimensions supérieures.

□

I.3 - ROTATION DE CONTRAINTES

Dans le but de rendre plus efficaces les méthodes algébriques de résolution des problèmes en nombres entiers, Kianfar [G40, G41] puis Walukiewicz et Kaliszewski [G62] proposent de transformer à chaque itération le problème en un problème équivalent (i.e. possédant le même domaine entier) dont le nombre de points extrêmes à coordonnées entières du domaine continu soit supérieur à celui du problème d'origine.

La méthode employée est la rotation de contraintes - soit celles du problème initial, soit les coupes générées en cours de méthode - de sorte que les nouvelles contraintes obtenues passent par au moins autant de points entiers que les originales.

La rotation d'une contrainte mise sous la forme

$$\sum_{j=1}^{n} a_j x_j \leq b \text{ où } b,\ a_j \in \mathbb{N} \quad j = 1,\ldots,n$$

est obtenue comme suit :

étant donné $k \in \{1,\ldots,n\}$, si la valeur de

$$(p^k) \quad \max \sum_{\substack{j=1 \\ j\neq k}}^{n} a_j x_j \text{ s.c. } \sum_{\substack{j=1 \\ j\neq k}}^{n} a_j x_j \leq b - a_k \ ;\ x_j = 0 \text{ ou } 1 \ \forall j \neq k$$

(dont une solution optimale est notée x^k) est strictement inférieure à $b - a_k$, alors il est clair que la contrainte :

$$\bar{a}_k \, x_k + \sum_{\substack{j=1 \\ j \neq k}}^{n} a_j \, x_j = b$$

où $\bar{a}_k = b - v(P^k)$

passera par le point entier $\underline{x}$ défini par :

$$\underline{x}_k = 1 \text{ et } \underline{x}_j = x_j^k \quad j = 1, \ldots, k-1, k+1, \ldots, n$$

ce qui n'était pas le cas pour la contrainte $\sum_{j=1}^{n} a_j \, x_j = b$.

Ce procédé peut alors être itéré sur une autre variable de la contrainte

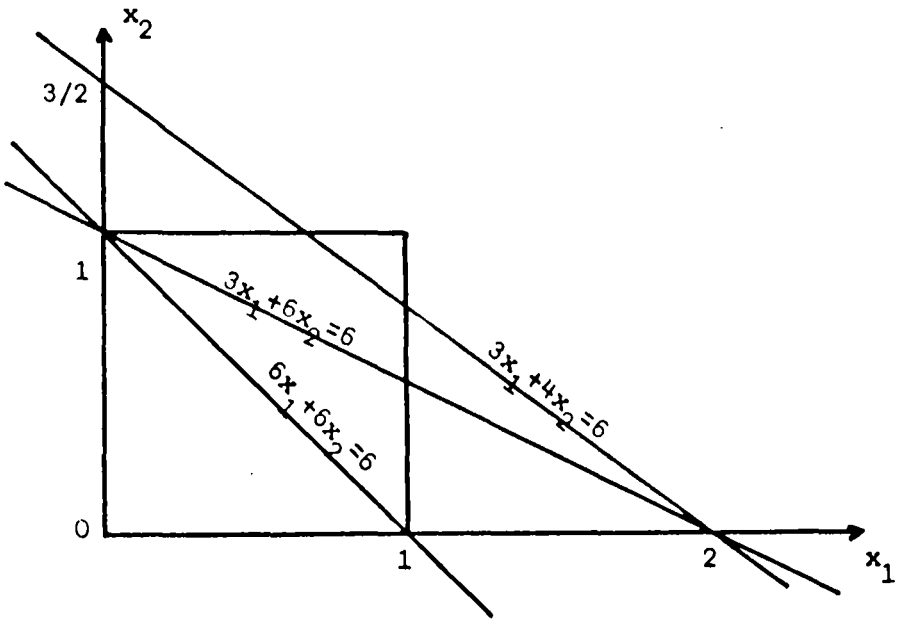
$$\bar{a}_k \, x_k + \sum_{\substack{j=1 \\ j \neq k}}^{n} a_j \, x_j \leq b$$

Note

Cette étude entre dans le cadre de celle faite par Bradley, Hammer et Wolsey [G6].

□

Exemple :



Les solutions bivalentes de $3x_1 + 4x_2 \leq 6$ sont (0,0), (1,0) et (0,1) ; l'application du principe exposé ci-dessus permet, en deux rotations successives, d'aboutir à la contrainte $6x_1 + 6x_2 \leq 6$ dont l'ensemble des solutions bivalentes est identique à la précédente, mais qui s'appuie sur les deux points entiers (0,1) et (1,0) :

$v(p^2) = \max \{3x_1 \mid 3x_1 \leq 6 - 4 \; ; \; x_1 = 0 \text{ ou } 1\} = 0 \quad \Longrightarrow \bar{a}_2 = 6$

$\Longrightarrow$ la contrainte initiale est transformée en $3x_1 + 6x_2 \leq 6$.

$v(p^1) = \max \{6x_2 \mid 6x_2 \leq 6 - 3 \; ; \; x_2 = 0 \text{ ou } 1\} = 0 \quad \Longrightarrow \bar{a}_1 = 6$

d'où la conclusion.

□

I.4 - CONTRACTION DE CONTRAINTES

I.4.1 - Introduction

L'élaboration de méthodes performantes de résolution du problème du knapsack ne peut qu'accroître l'intérêt des résultats généralisant et améliorant de façon sensible ceux de Mathews [E8] qui en 1897 avait démontré qu'il est possible de transformer un système d'équations diophantiennes à coefficients entiers tous strictement positifs en une équation unique équivalente (i.e. dont l'ensemble des solutions entières réalisables est identique à celui du système).

De nombreux auteurs (Anthonisse [E1], Bradley [E2], Glover [E4], Glover et Woolsey [E5], Padberg [E10], Zionts [G64], dans les années 70-72, et plus récemment Kaliszewski et Libura [E6], Kendall et Zionts [E7], Meyer [E9] et Weinberg [E11, E12] dans les années 76-77) se sont donc intéressés au problème de la *contraction* de m contraintes en une seule, c'est-à-dire à celui de remplacer un système de la forme

$$(S_1) \quad \begin{cases} Ax = b \\ x \in C \cap \mathbb{N}^n \end{cases} \qquad \text{où} \quad \begin{cases} A \text{ est le format } (m \times n) \\ b \text{ est le format } (m \times 1) \\ a_{ij}, b_i \in \mathbb{Z} \quad \forall i \; \forall j \\ C \subset \mathbb{R}^n \end{cases}$$

par le système équivalent

$$(S_2) \quad \begin{cases} \alpha\, Ax = \alpha b \quad (10) \\ x \in C \subset \mathbb{N}^n \end{cases} \qquad \text{où } \alpha \in \mathbb{Z}^m$$

Avant d'exposer le principe des méthodes de contraction de deux contraintes en une seule, il est indispensable de noter que l'équation (10) est obtenue à la suite d'une cascade de combinaisons linéaires d'équations prises deux à deux. Ses coefficients, qui sont donc fonctions à la fois de la taille du système (S_1) (nombre de contraintes et de variables) et des valeurs des coefficients des contraintes initiales, peuvent prendre des valeurs extrêmement grandes. Même pour des systèmes de petite taille mais à coefficients élevés, la manipulation d'entiers aussi grands peut conduire à des difficultés sur le plan informatique.

Par contre, un système de contraintes n'admettant pour éléments non nuls que des coefficients égaux à un, peut éventuellement se prêter plus favorablement à l'application de telles méthodes, et ne pas conduire à des coefficients excessifs (pour fixer les idées, citons l'exemple de 22 contraintes d'un problème de 177 variables [G38] possédant environ $7.3^{}$ d'éléments égaux à 1 qui a abouti à des coefficients non négatifs inférieurs à $3.5\ 10^{12}$, en utilisant les résultats de [E2]).

Ce sera l'objet de l'étude faite au chapitre 3, partie II.

I.4.2 - Principe des méthodes

Puisque la contraction de plus de deux contraintes en une seule se déduit de celle de deux contraintes en une seule par itération du procédé, l'étude suivante concerne l'équivalence de

$$\left|\begin{array}{l} A_1\ x = b_1 \\ A_2\ x = b_2 \\ x \in X \cap C \subset \mathbb{N}^n \end{array}\right.$$

et de $\left|\begin{array}{l} (\alpha_1 A_1 + \alpha_2 A_2)\ x = \alpha_1 b_1 + \alpha_2 b_2 \\ x \in X \end{array}\right.$

En notant $\phi_i : X \to Z$

$$x \mapsto \phi_i(x) = b_i - A_i\ x \qquad \text{pour } i = 1 \text{ et } 2$$

le théorème 6 fournit le résultat le plus utilisé dans la littérature :

Théorème 6

Le système

$$(S_1) \quad \left|\begin{array}{l} \phi_1(x) = 0 \\ \phi_2(x) = 0 \\ x \in X \end{array}\right.$$

est équivalent à

$$(S_2) \quad \left|\begin{array}{l} \alpha_1\ \phi_1(x) + \alpha_2\ \phi_2(x) = 0 \\ x \in X \end{array}\right.$$

sous les hypothèses suivantes

(i) $\alpha_1, \alpha_2 \in \mathbb{N}^*$: P.G.C.D. $(\alpha_1, \alpha_2) = 1$

(ii) $\alpha_1 > \max_{x \in X} |\phi_2(x)|$ *ou* $\alpha_2 > \max_{x \in X} |\phi_1(x)|$

Démonstration

(i) $\forall x \in X : \phi_1(x) = \phi_2(x) = 0 \Rightarrow \alpha_1\ \phi_1(x) + \alpha_2\ \phi_2(x) = 0$

$\forall\ \alpha_1, \alpha_2 \in \mathbb{R}$

(ii) Etant donné $x \in X$ tel que

$$\begin{cases} \alpha_1 \phi_1(x) + \alpha_2 \phi_2(x) = 0 & (11) \\ \alpha_2 > \max\limits_{x \in X} |\phi_1(x)| & (12) \\ \text{P.G.C.D. } (\alpha_1, \alpha_2) = 1 & (13) \end{cases}$$

On montre que l'hypothèse $\phi_2(x) \neq 0$ est contradictoire :

$$(11) \Rightarrow \phi_1(x) = -(\alpha_2/\alpha_1)\,\phi_2(x) \Rightarrow |\phi_1(x)| = (\alpha_2/\alpha_1)|\phi_2(x)| \quad (14)$$
$$|\phi_1(x)|, |\phi_2(x)| \in \mathbb{N} \quad \text{et} \quad (13)$$

$$\ldots\ldots \Rightarrow \begin{matrix} |\phi_2(x)|/\alpha_1 \in \mathbb{N} \\ \alpha_1 \in \mathbb{N}^*, \ \phi_2(x) \neq 0 \end{matrix} \Rightarrow \underset{(14)}{|\phi_2(x)|/\alpha_1 \geq 1} \Rightarrow |\phi_1(x)| \geq \alpha_2$$

ce qui est impossible.

$$\text{Ainsi } \forall x \in X \ (11), (12) \text{ et } (13) \Rightarrow \underset{(11)}{\phi_2(x) = 0} \quad \ldots\ldots$$

$$\ldots\ldots \Rightarrow \begin{matrix} \alpha_1 \phi_1(x) = 0 \\ \alpha_1 \neq 0 \end{matrix} \Rightarrow \phi_1(x) = 0.$$

□

Une généralisation de ce résultat a été récemment formulé par Kendall et Zionts [E7] :

Théorème 7

Les systèmes (S_1) et (S_2) sont équivalents sous les hypothèses suivantes :

(i) P.G.C.D. $(\alpha_1, \alpha_2) = 1$

(ii) $\forall x \in X$ α_1 *ne divise pas* $\phi_2(x)$ *ou* α_2 *ne divise pas* $\phi_1(x)$.

Démonstration

Etant donnés α_1 et α_2 vérifiant (i) et (ii),et $x \in X$ tel que $\alpha_1 \phi_1(x) + \alpha_2 \phi_2(x) = 0$, il faut montrer que $\phi_1(x) = \phi_2(x) = 0$ (la réciproque est triviale).

$$\left.\begin{array}{l} (11) \Rightarrow \phi_1(x) = -(\alpha_2/\alpha_1)\ \phi_2(x) \quad (15) \\ \qquad\qquad \phi_1(x) \in \mathbb{Z} \end{array}\right\} \Rightarrow -(\alpha_2/\alpha_1)\ \phi_2(x) \in \mathbb{Z} \left.\begin{array}{l} \\ \text{P.G.C.D. } (\alpha_1, \alpha_2) = 1 \end{array}\right\} \ldots.$$

$$\left.\begin{array}{l} \ldots. \Rightarrow \phi_2(x)/\alpha_1 \in \mathbb{Z} \\ \qquad (15) \end{array}\right\} \quad \ldots. \Rightarrow \phi_1(x)/\alpha_2 \in \mathbb{Z}, \text{ ce qui contredit (ii)}$$

sauf si $\phi_1(x) = \phi_2(x) = 0$.

□

Note

Bien que les problèmes

$$(P) \qquad \left[\begin{array}{l} \max \ cx \\ \text{s.c. } Ax = b \\ \qquad x \in C \cap \mathbb{N}^n \end{array}\right.$$

et

$$(P(\alpha)) \qquad \left[\begin{array}{l} \max \ cx \\ \text{s.c. } \alpha Ax = \alpha b \\ \qquad x \in C \cap \mathbb{N}^n \end{array}\right.$$

possèdent la même valeur, il n'en est pas de même pour $(\bar{P})$ et $(\overline{P(\alpha)})$. Puisque $(\overline{P(\alpha)})$ est une relaxation de $(\bar{P})$, la double inégalité suivante est vérifiée :

$$v(P) \leq v(\bar{P}) \leq v(\overline{P(\alpha)}) \qquad (16)$$

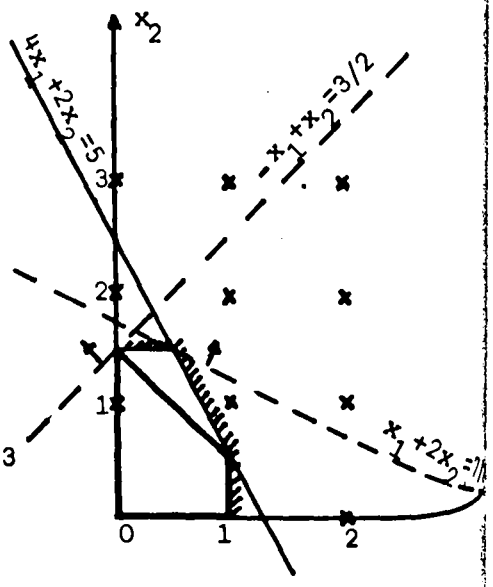
Pour les problèmes traités au chapitre 3, on aura toujours $v(\overline{P(\alpha)}) >> v(\bar{P})$; mais ce résultat n'est pas général comme le prouvent les exemples suivants pour lesquels on retrouve l'éventail des quatre possibilités offertes par la relation (16) :

Exemple 1

Soit le système de contraintes

$$(I)\begin{cases} 2x_1 + 2x_2 \leq 3 \\ x_1 \leq 1 \\ x_1, x_2 \geq 0 \end{cases}$$

(domaine limité par les traits gras).



$$(I) \Leftrightarrow \begin{cases} 2x_1 + 2x_2 + x_3 = 3 \\ x_1 + x_4 = 1 \\ x_i \geq 0 \quad i = 1,\ldots,4 \end{cases} \Rightarrow \begin{cases} x_2 \leq 3/2, x_3 \leq 3 \\ x_1 \leq 1, x_4 \leq 1 \end{cases}$$

Le système diophantien associé à (I)

$$(I')\begin{cases} 2x_1 + 2x_2 + x_3 = 3 \quad (17) \\ x_1 + x_4 = 1 \quad (18) \\ x_i \leq 1, x_2 \leq 3/2, x_3 \leq 3, x_4 \leq 1 \\ x_i \in \mathbb{N} \quad i = 1,\ldots,4 \end{cases}$$

admet après contraction des contraintes (17) et (18) ($\alpha_1 = 1$ et $\alpha_2 = 2$), le système équivalent suivant :

$$(II') \quad \begin{cases} 4x_1 + 2x_2 + x_3 + 2x_4 = 5 \\ x_1 \leq 1, x_2 \leq 3/2, x_3 \leq 3, x_4 \leq 1 \\ x_i \in \mathbb{N},\ i = 1,\ldots,4 \end{cases}$$

Le système en variables continues associé est donc :

$$(II) \quad \begin{cases} 4x_1 + 2x_2 \leq 5 \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_2 \leq 3/2 \end{cases}$$

(domaine limité par les traits hachurés).

Pour cet exemple et le suivant, on a donc

(P) max cx s.c. x vérifie (II)
$(\bar{P})$ max cx s.c. x vérifie (I)
$(\overline{P(\alpha)})$ max cx s.c. x vérifie (II').

a) $c = [-1, 1] \Rightarrow v(\overline{P(\alpha)}) = v(\bar{P}) = 3/2 > v(P) = 1$

b) $c = [1, 2] \Rightarrow v(\overline{P(\alpha)}) = 7/2 > v(\bar{P}) = 3 > v(P) = 2.$

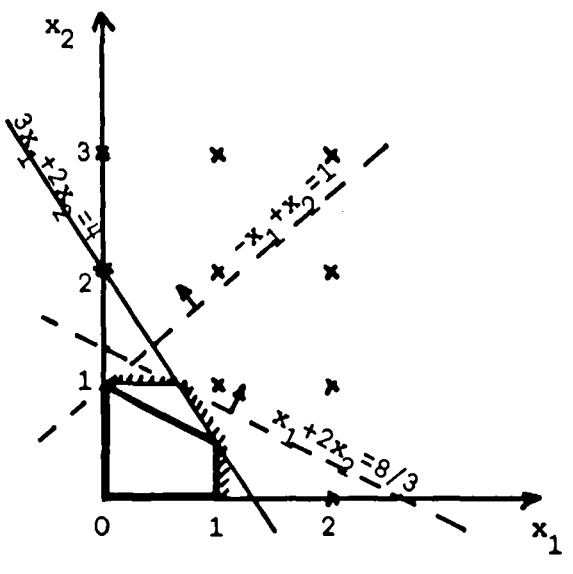
□

Exemple 2

Soit le système de contraintes

$$(I) \quad \begin{cases} x_1 + 2x_2 \leq 2 \\ x_1 \leq 1 \\ x_1, x_2 \geq 0 \end{cases}$$

(domaine limité par un trait gras).



$$(I)\quad \begin{cases} x_1 + 2x_2 + x_3 = 2 \\ x_1 + x_4 = 1 \\ x_i \geq 0 \quad i = 1,\ldots,4 \end{cases} \Longrightarrow \begin{cases} x_2 \leq 1, x_3 \leq 2 \\ x_1 \leq 1, x_4 \leq 1 \end{cases}$$

Le système diophantien associé à (I)

$$(I')\quad \begin{cases} x_1 + 2x_2 + x_3 = 2 \quad (19) \\ x_1 + x_4 = 1 \quad (20) \\ x_1 \leq 1, x_2 \leq 1, x_3 \leq 2, x_4 \leq 1 \\ x_i \in \mathbb{N} \quad i = 1,\ldots,4 \end{cases}$$

admet après contraction des contraintes (19) et (20) ($\alpha_1 = 1$ et $\alpha_2 = 2$), le système équivalent suivant :

$$(II')\quad \begin{cases} 3x_1 + 2x_2 + x_3 + 2x_4 = 4 \\ x_1 \leq 1, x_2 \leq 1, x_3 \leq 2, x_4 \leq 1 \\ x_i \in \mathbb{N} \quad i = 1,\ldots,4 \end{cases}$$

Le système en variables continues associé est donc le suivant :

$$(II)\quad \begin{cases} 3x_1 + 2x_2 \leq 4 \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_2 \leq 1 \end{cases}$$

(domaine limité par les traits hachurés).

a) $c = [-1, 1] \Longrightarrow v(\overline{P(\alpha)}) = v(\bar{P}) = v(P) = 1$

b) $c = [1, 2] \Rightarrow v(\overline{P(\alpha)}) = 8/3 > v(\bar{P}) = v(P) = 2.$

□

II - RELAXATION DU KNAPSACK

La solution de la relaxation de (B) obtenue en relâchant les conditions d'intégrité sur les variables

$$(\bar{B}) \quad \begin{cases} \max \ cx \\ \text{s.c. } ax \leq b \\ \qquad x \in [V] \end{cases}$$

est fournie par le

Théorème 8

Etant donné $U \subset K = \{j \in I = \{1,\ldots,n\} \mid a_j \leq b\}$ *tel que*

$$\begin{cases} \sum_{j \in U} a_j \leq b < \sum_{j \in U \cup \{i\}} a_j \\ \forall j \in U \ c_j/a_j \geq c_p/a_p \quad \forall p \notin U \\ c_i/a_i = \max \{c_p/a_p \mid p \notin U\} \end{cases}$$

une solution optimale $\bar{x}$ *de* $(\bar{B})$ *est définie par*

$$\begin{cases} \bar{x}_j = 1 & \forall j \in U \\ \bar{x}_i = (b - \sum_{j \in U} a_j) \ / \ a_i \\ \bar{x}_j = 0 & \forall j \in L = I \setminus (U \cup \{i\}). \end{cases}$$

Démonstration : Evidente dans ce cas particulier de la programmation linéaire.

□

Note

(i) lorsque $\sum_{j\in U} a_j = b$, $\bar{x} \in V(\bar{x}_j = 1 \ \forall j \in U, \ \bar{x}_j = 0 \ \forall j \in L$ =

$\Longrightarrow x^* = \bar{x}$

(ii) la restriction de I à K permet de réduire la valeur de $(\bar{B})$ lorsque $\exists j \in I \backslash K \quad c_j/a_j > c_i/a_i$.

□

Corollaire

(i) $\underline{v}(B) \geq \sum_{j\in U} c_j \ ; \ \bar{v}(B) \leq \lfloor v(\bar{B}) \rfloor$

(ii) $\bar{v}(B) - \underline{v}(B) < c_i$ (21)

Démonstration

* . $\underline{x}=[\bar{x}] \in V : \underline{x}_j = 1 \ \forall j \in U, \ \underline{x}_j = 0 \ \forall j \notin U$ est une solution réalisable de (B) ; la valeur $c\,\underline{x}$ obtenue ainsi est à l'évidence un minorant de v(B).

. $v(B) \leq v(\bar{B})$ et $v(B) \in \mathbb{N} \Longrightarrow \lfloor v(\bar{B}) \rfloor$ est un majorant de v(B)

* (i) $\Rightarrow \bar{v}(B) - \underline{v}(B) \leq \lfloor v(\bar{B}) \rfloor - \sum_{j\in U} c_j \leq v(\bar{B}) - \sum_{\substack{j\in U \\ \bar{x}_i < 1}} c_j = c_i \bar{x}_i \Rightarrow$ (

□

Jusqu'en 1977, ($\bar{B}$) a toujours été résolu par l'algorithme classique - que nous appelerons C.K.R. (i.e. Classical Knapsack Relaxation) - exposé en figure 1.

---

1 Trier les données pour obtenir (après renumérotation des variables)

$$\frac{c_{j_1}}{a_{j_1}} \geq \frac{c_{j_2}}{a_{j_2}} \geq \ldots \geq \frac{c_{j_n}}{a_{j_n}} \qquad (22)$$

2 Poser $k^* = \min \{k \mid \sum_{\substack{p=1 \\ p \in K}}^{k} a_{j_p} > b\}$

3 Poser $U = \{j_1, \ldots, j_{k^*-1}\} \cap K$

si $\sum_{j \in U} a_j = b$ alors $L \equiv I \setminus U$ sinon $i \leftarrow j_{k^*}$, $L \equiv I \setminus (U \cup \{i\})$, $\bar{x}_i \leftarrow (b - \sum_{j \in U} a_j) / a_i$

4 $\bar{x}_j \leftarrow 1 \ \forall j \in U$, $\bar{x}_j \leftarrow 0 \ \forall j \in L$; $v(\bar{B}) \leftarrow c\,\bar{x}$

Figure 1 : Algorithme C.K.R.

Compte-tenu du temps de calcul prohibitif de la première étape de cet algorithme (voir partie VII et les références [K12 , K13]), nous proposons de déterminer U (et i s'il y a lieu) sans avoir recours au tri systèmatique des données nécessaire à l'obtention de (22).

Le nouvel algorithme envisagé pour la résolution de ($\bar{B}$) - appelé N.K.R. (i.e. New Knapsack Relaxation) - est une adaptation de la méthode déterminant les p plus grands éléments d'une liste de n nombres (p donné < n), qui est elle-même dérivée de l'algorithme de tri Quicksort (Voir [C1, C2, C9, C15, C17, C21]).

L'exposé de son principe (figure 2) nécessite la donnée de la :

Définition 13

Etant donnés $\begin{cases} k \in J \subset Q \subset \mathbb{N} \\ Z = \bigcup_{j \in Q} \{z_j\} \text{ où } z_j \in \mathbb{R} \quad \forall j \in Q \end{cases}$

la (Z, k) - partition de J est définie par la réunion des trois sous-ensembles de J

$$U(J, Z, k) = \{j \in J \mid z_j > z_k\}$$
$$E(J, Z, k) = \{j \in J \mid z_j = z_k\}$$
$$L(J, Z, k) = J \setminus (U(J, Z, k) \cup E(J, Z, k)).$$

et du

Théorème 9

$$\textit{Etant donnés} \quad \begin{cases} r \in \mathbb{R} \text{ et } E = \{j \in I \mid c_j = r\, a_j\} \\ d \leq b \end{cases}$$

la résolution de

$$\max \sum_{j \in E} c_j x_j \text{ s.c. } \sum_{j \in E} a_j x_j \leq d \; ; \; 0 \leq x_j \leq 1 \quad \forall j \in E$$

se réduit à la détermination de $G \subset E$ *tel que*

$$\sum_{j \in G} a_j \leq d \text{ et } \exists p \in E \setminus G \; d < \sum_{j \in G \cup \{p\}} a_j$$

Démonstration : Voir lemme du théorème 1 de [K13].

□

0 $J \equiv K$; $U \equiv \emptyset$; $R \equiv \bigcup_{j \in J} \{c_j/a_j\}$

1 Sélectionner $k \in J$ et déterminer la (R,k) - partition de J

2 si $\sum_{j \in U(J,R,k)} a_j > b$ alors $J \equiv U(J, R, k)$; aller en 1
sinon $U \equiv U \cup U(J, R, k)$

3 si $\sum_{j \in U} a_j = b$ alors $L \equiv I \backslash U$; aller en 6

4 si $b < \sum_{j \in U} a_j + a_k$ alors $i \leftarrow k$; $L \equiv I \backslash (U \cup \{i\})$
$\bar{x}_i \leftarrow (b - \sum_{j \in U} a_j)/a_i$
aller en 6

5 si $\sum_{j \in U \cup E(J,R,k)} a_j \geq b$

alors construire un sous-ensemble G de E(J, R, k)
tel que
$$\sum_{j \in U \cup G} a_j \leq b < \sum_{j \in U \cup G} a_j + a_p, \quad p \in I \backslash (U \cup G)$$

$U \equiv U \cup G$;
si $\sum_{j \in U} a_j = b$ alors $L \equiv I \backslash U$ sinon $i \leftarrow p$; $L \equiv I \backslash (U \cup \{i\})$
$\bar{x}_i \leftarrow (b - \sum_{j \in U} a_j)/a_i$
aller en 6

sinon $U \equiv U \cup E(J, R, k)$; $J \equiv L(J, R, k)$
aller en 1

6 $\bar{x}_j \leftarrow 1 \quad \forall j \in U$, $\bar{x}_j \leftarrow 0 \quad \forall j \in L$; $v(\bar{B}) \leftarrow c\bar{x}$; stop

Figure 2 : Algorithme N.K.R.

Exemple numérique : problème de 12 variables tiré de [K6]

j	:	1	2	3	4	5	6	7	8	9	10	11	12
c_j	:	10	10	13	7	5	14	7	6	13	15	8	10
a_j	:	7	9	10	12	5	7	10	13	7	10	6	8
c_j/a_j	:	1.43	1.11	1.33	.58	1	2	.70	.46	1.86	1.50	1.30	1.25

, $b = 51$

Itération 1 :

$$J_1 = \{1, 2, \ldots, 12\},\ k_1 = 1,\ U = \emptyset$$

(R,k_1)-partition de J_1 :

j	:	10	9	6	1	5	4	7	8	3	2	11	12
c_j	:	15	13	14	10	5	7	7	6	13	10	8	10
a_j	:	10	7	7	7	5	12	10	13	10	9	6	8

$U(J_1,R,k_1) = \{10, 9, 6\}$, $\sum_{j \in U(J_1,R,k_1)} a_j + a_{k_1} = 31 < b = 51$

Itération 2 :

$U = \{10, 9, 6, 1\}$, $J_2 = L(J_1,R,k_1) = \{5, 4, 7, 8, 3, 2, 11, 12\}$, $b = 20$, $k_2 = 5$

(R,k_2)-partition de J_2 :

j	:	12	11	2	3	5	8	7	4
c_j	:	10	8	10	13	5	6	7	7
a_j	:	8	6	9	10	5	13	10	12

$U(J_2,R,k_2) = \{12, 11, 2, 3\}$, $\sum_{j \in U(J_2,R,k_2)} a_j = 33 > b = 20$

Itération 3 :

$U = \{10, 9, 6, 1\}$, $J_3 = U(J_2,R,k_2) = \{12, 11, 2, 3\}$, $b = 20$, $k_3 = 12$

(R,k_3)-partition de J_3 :

j	:	3	11	12	2
c_j	:	13	8	10	10
a_j	:	10	6	8	9

$U(J_3,R,k_3) = \{3,11\}$, $\sum_{j \in U(J_3,R,k_3)} a_j = 16 < b < \sum_{j \in U(J_3,R,k_3)} a_j + a_{k_3} = 24$

$\Longrightarrow U = \{10, 9, 6, 1, 3, 11\}$, $i = 12$, $L = \{2, 5, 8, 7, 4\}$

Ainsi le fichier des c_j/a_j obtenu en fin d'algorithme NKR est le suivant :

c_j	:	15	13	14	10	13	8	10	10	5	6	7	7
a_j	:	10	7	7	7	10	6	8	9	5	13	10	12

alors que ce fichier classé suivant l'ordre décroissant des c_j/a_j est le suivant :

c_j	:	14	13	15	10	8	13	10	10	5	7	7	6
a_j	:	7	7	10	7	6	10	8	9	5	10	12	13

Les justifications et différentes variantes de l'algorithme N.K.R [au niveau de l'étape 1] sont données dans [K13] où il est également démontré le

Théorème 10

L'algorithme N.K.R est de complexité $O(|K|)$ en moyenne.

□

L'algorithme exposé ici est en de nombreux points semblable à celui de Balas et Zemel [K2] ; par contre celui de Lawler [A3] diffère un peu au niveau du principe de l'étape 1 :
au lieu de la séquence

(i) sélection de $k \in J$
(ii) détermination de la (R,k)-partition de J

l'auteur propose la détermination de la *médiane* de $R = \bigcup_{j \in J} \{c_j/a_j\}$ pour lequel il existe un algorithme de complexité linéaire (voir [C1]), ce qui lui permet d'aboutir à une complexité globale $O(|K|)$ dans le cas le plus défavorable puis que le nombre d'opérations sera borné par

$$h\ |K| \sum_{j=0}^{\log_2 |K|} (1/2^j) < 2\ h\ |K| \text{ où } h \in \mathbb{N}.$$

De nouvelles variantes de notre algorithme sont proposées en partie VII ; on verra en particulier que la meilleure d'entre elles apporte en moyenne un gain de 82 % par rapport à l'algorithme C.K.R. pour les problèmes de 9 000 yariables tirés au hasard.

III - METHODES D'ENCADREMENT

L'utilisation de méthodes d'encadrement de la fonction économique et de la contrainte va permettre de resserrer le domaine initial de (B) qui deviendra

$$F(B) = \{x \in V \mid \underline{v}(B) < cx \leq \bar{v}(B),\ \underline{b} \leq ax \leq b\}.$$

Les déterminations de $\underline{v}(B)$, $\bar{v}(B)$ et $\underline{b}$ (§ III.1, III.2, et III.3) sont suivies de leur utilisation dans le cadre de l'évaluation du nombre maximum de solutions réalisables de (B) (§ III.4).

III.1 - MINORANTS DE v(B)

L'obtention d'un minorant de la valeur de (B) peut se faire par l'intermédiaire de nombreux procédés appelés *heuristiques* ; celles-ci se répartissent, dans le cadre de la D-analyse, en trois classes dont les algorithmes les plus performants seront détaillés :

- les algorithmes *gloutons* (sept méthodes sont décrites)
- les algorithmes *approchants polynômiaux* (Johnson [A7], Sahni [A12] et Chandra, Hirschberg et Wong [A1])
- les algorithmes *approchants totalement polynômiaux* (Ibarra et Kim [A6], Lawler [A9] et Magazine, Oguz [A11]) dont l'exposé nécessitera des rappels sur la *programmation dynamique* ([G5, G56, G63, K20]).

Note

Se référer à [A2] pour une P-analyse d'un algorithme glouton.

□

III.1.1 - Algorithmes gloutons

L'algorithme glouton est le type d'heuristique utilisé le plus couramment dans le cadre de la résolution des problèmes d'optimisation en nombres entiers.

Si le problème possède n variables, un tel algorithme fournit une solution réalisable en fixant tour-à-tour les valeurs des variables (n étapes au maximum sont nécessaires) ; chaque fixation est faite dans le but d'obtenir une solution améliorant localement la solution précédente.

III.1.1.1 - D-analyse

Dans le cas du problème du knapsack, présenté ici dans son cadre le plus général :

$$(B)\quad \left[\begin{array}{l} \max \sum_{j=1}^{n} c_j x_j \\ \text{s.c.} \sum_{j=1}^{n} a_j x_j \leq b \\ \qquad 0 \leq x_j \leq u_j \\ \qquad x_j \in \mathbb{N} \quad j = 1,\ldots,n \end{array}\right.$$

$$\text{où}\quad \left[\begin{array}{l} a_j \leq b \quad j = 1,\ldots,n \\ u_j = \begin{cases} \lfloor b/a_j \rfloor & \text{si le problème est non borné} \\ q_j < \lfloor b/a_j \rfloor & \text{sinon } (q_j = 1 \text{ pour le knapsack 0 - 1}) \end{cases} \end{array}\right.$$

le type d'algorithme glouton - appelé *algorithme G* - le plus simple est le suivant (il fournit une solution réalisable $\underline{x}$ de (B) et la valeur associée $v_G(B)$) :

Algorithme G

1 $j \leftarrow 1$; $\bar{b} \leftarrow b$; $v_G(B) \leftarrow 0$;

2 $x_j \leftarrow \min(u_j, \lfloor \bar{b}/a_j \rfloor)$; $\bar{b} \leftarrow \bar{b} - a_j x_j$;
$v_G(B) \leftarrow v_G(B) + c_j x_j$;

3 si $j = n$ alors stop ;

4 si $\bar{b} = 0$ alors $x_k \leftarrow 0$ $k = j+1,\ldots,n$; stop

5 $j \leftarrow j + 1$; aller en 2 ;

Notes

(i) l'algorithme G est de complexité $O(n)$

(ii) en général l'algorithme G est précédé d'un tri des données

- s'il s'agit d'un classement par ordre décroissant suivant les valeurs des c_j/a_j $j = 1,\ldots,n$, alors il est clair que globalement l'algorithme atteint la complexité $O(n \log_2 n)$.

- s'il s'agit par contre du résultat fourni par l'algorithme N.K.R. alors la complexité globale de *l'algorithme glouton* précédé du tri reste égale à $O(n)$; cet algorithme sera noté par la suite G^*, sa valeur sera donc notée $v_{G^*}(B)$, pour tout problème du knapsack (B) envisagé.

□

Les résultats suivants prouvent que l'algorithme glouton est 1/2-approchant dans le cas du problème du knapsack non borné (i.e. $u_j = \lfloor b/a_j \rfloor$ $\forall j$), alors que dans le cas du knapsack borné (par exemple, $u_j = 1$ $\forall j$) il n'existe aucun ε de $[0,1[$ pour lequel il soit ε-approchant.

Théorème 11 (Fisher [A3], Kannan et Korte [A8])

Lorsque

$$c_1/a_1 = \max_{1\le j\le n} (c_j/a_j) \qquad (23)$$

l'algorithme G est 1/2-approchant pour le problème du knapsack non bor

Démonstration

Il faut montrer deux résultats sous l'hypothèse (23)

(i) $2\, v_G(B) \ge v(B)$

(ii) $\exists (E_k)_{k\in\mathbb{N}}$ suite d'exemplaires de (B) : $v_G(E_k)/v(E_k) \underset{k\to\infty}{\to} 1/2$

(i) par hypothèse $u_j = \lfloor b/a_j \rfloor \ \forall j \Rightarrow v(\bar{B}) = c_1 b/a_1$ (24)

(puisque le dual de $(\bar{B})$: min ub s.c. ua > c ; u ≥ 0
admet pour valeur $b \max_{1\le j\le n} (c_j/a_j)$)

$v(B) \le v(\bar{B})$ et (24)
$v_G(B) \ge c_1 \lfloor b/a_1 \rfloor$, par construction $\Big\} \Rightarrow v_G(B)/v(B) \ge \lfloor b/a_1 \rfloor / (b/$
$b/a_1 \le \lfloor b/a_1 \rfloor + 1$

$\ldots\Rightarrow$ $v_G(B)/v(B) \ge \lfloor b/a_1 \rfloor / (\lfloor b/a_1 \rfloor + 1)$
$\forall j\ a_j \le b \Rightarrow \lfloor b/a_1 \rfloor \ge 1$ $\Big\} \Rightarrow 2\, v_G(B) \ge v(B).$

(ii) On considère les exemplaires $(E_k)_{k\ge 2}$ à deux variables suivants :

$$\max (k+2)x_1 + k\, x_2 \text{ s.c. } (k+1)x_1 + k\, x_2 \le 2k \ ;\ x_1, x_2 \in \mathbb{N}$$

$\Rightarrow$ $V_G(E_k) = k + 2$ et $v(E_k) = 2k \Rightarrow v_G(E_k)/v(E_k) = (k+2)/2k \underset{k\to\infty}{\to} 1/2$.

□

Note

Une conséquence évidente du théorème 11 est : l'algorithme G est ε-approchant $\forall\ \varepsilon \in [1/2,1[$.

□

Théorème 12 (Fisher [A3], Kannan et Korte [A8])

Dans le cas du problème du knapsack en variables 0-1, l'algorithme G n'est ε-approchant pour aucun ε de $[0,1[$, quelque soit la relation d'ordre imposée aux données.

Démonstration

Pour chacune des hypothèses de tri des données, il suffit de trouver une suite d'exemplaires (E_k) de (B) pour lesquels

$$v_G(E_k) \;/\; v(E_k) \underset{k\to+\infty}{\to} 0$$

On ne retiendra dans cette démonstration que les deux hypothèses de tri, les plus usitées : l'ordre décroissant des c_j/a_j et celui des c_j $j = 1,\ldots,n$:

(i) *hypothèse* : $c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n$.

On envisage la suite de problèmes $(E_k)_{k\geq 2}$ à 2 variables

$$\max\ 2x_1 + kx_2 \ \text{s.c.}\ x_1 + k\,x_2 \leq k\ ;\ x_1,\ x_2 = 0 \text{ ou } 1$$

$$\Rightarrow\ v_G(E_k) = 2 \text{ et } v(E_k) = k \Rightarrow v_G(E_k)/v(E_k) = 2/k \underset{k\to+\infty}{\to} 0$$

(ii) *hypothèse* : $c_1 \geq c_2 \geq \ldots \geq c_n$.

Les exemplaires $(E_k)_{k\geq 2}$ de (B) envisagés sont les problèmes à k variables suivants :

$$(E_k)\quad \begin{cases} \max\ k\,x_1 + (k-1)\,x_2 + \ldots + (k-1)\,x_k \\ s.c.\ k\,x_1 + x_2 + \ldots + x_k \le k \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,k \end{cases}$$

$\Rightarrow\ v_G(E_k) = k$ et $v(E_k) = (k-1)^2$

$\Rightarrow\ v_G(E_k)/v(E_k) = k/(k-1)^2 \to 0$ $\quad k \to +\infty$.

□

Note

Si le problème en variables 0-1 est dérivé d'un problème en nombres entiers par multiplication du nombre de variables du type $x_j = \sum_{j=1}^{\lfloor b/a_j \rfloor} y_{jk}$ avec $y_{jk} = 0$ ou 1, alors l'algorithme G est 1/2-approc d'après le théorème 11.

□

III.1.1.2 - Catalogue

Les sept heuristiques $H_j (j = 1,\ldots,7)$ de la littérature - de valeur $\lambda_j = v_j(B)$ et de solution x^j - sont cataloguées ci-dessous dans l'ordre chronologique.

Après avoir noté que

$$v_k(B) \ge \sum_{j \in U} c_j \quad k = 1,\ldots,7,$$

une mesure de leurs précisions, *dépendante* des données du problème, est fournie par le résultat suivant (à rapprocher avec celui de [K44]) :

Théorème 13

$$\forall\ \lambda \in \left[\sum_{j \in U} c_j,\ v(B) \right]$$

$$(v(B) - \lambda) / v(B) \leq 1 - \left(\sum_{j \in U} a_j / b \right)$$

Démonstration

$$\forall \lambda \in \left[\sum_{j \in U} c_j, v(B) \right]$$

$$\left.\begin{aligned} (v(B)-\lambda)/v(B) &\leq (v(B) - \sum_{j \in U} c_j)/v(B) \\ &\leq (v(\bar{B}) - \sum_{j \in U} c_j)/v(B) = \frac{(c_i/a_i)\,(b - \sum_{j \in U} a_j)}{\sum_{j \in U} c_j + (c_i/a_i)(b - \sum_{j \in U} a_j)} \end{aligned}\right\} \quad \ldots.$$

$$\left.\begin{aligned} &\forall j \in U \quad c_j \geq (c_i/a_i)\, a_j \Rightarrow \sum_{j \in U} c_j \geq (c_i/a_i) \sum_{j \in U} a_j \\ &\ldots. \Rightarrow (v(B)-\lambda)/v(B) \leq \frac{(c_i/a_i)\,(b - \sum_{j \in U} a_j)}{(c_i/a_i)b} = 1 - \left(\sum_{j \in U} a_j / b \right). \end{aligned}\right\}$$

□

H_1 - Dantzig (1957) [K5]

Suivant l'ordre p dans lequel les variables sont prises en compte :

- décroissant des rapports c_j/a_j (p=1), des c_j(p=2), des $c_j - a_j$(p=3)
- croissant des rapport c_j/a_j (p = 4), des a_j (p = 5)
- naturel des variables (p = 6)

cette méthode aboutit à la valeur

$$v_{1.p}(B) = \lambda_{1.p} = \sum_{j=1}^{k_p - 1} c_j \text{ avec } k_p = \min \left\{ k \mid \sum_{j=1}^{k} a_j > b \right\}$$

déterminée par l'algorithme suivant :

1 $\lambda_{1.p} \leftarrow 0$; $j \leftarrow 1$;

2 Si $a_j \leq b$ alors $\lambda_{1.p} \leftarrow \lambda_{1.p} + c_j$;
$b \leftarrow b - a_j$
Si $j < n$ alors $j \leftarrow j + 1$; aller en 2 ;

3 $k_p \leftarrow j$; stop

Note

Lorsque les variables sont classées suivant l'ordre décroissant des c_j/a_j (i.e. p = 1) alors :

$k_p = i$, $x^{1.1} = [\bar{x}]$.

Ce classement sera adopté pour les heuristiques H_j j = 2,...,6.

□

H_2 - Greenberg et Hegerich (1970) [K23]

La première phase de la méthode de résolution exacte de (B) établie par Greenberg et Hegerich, fournit une solution $x^2 \in F(B)$ définie par

$$x_j^2 = \begin{cases} 1 \text{ si } a_j \leq b - \sum_{k=1}^{j-1} a_k x_k^2 \\ 0 \text{ sinon} \end{cases} \qquad j = 1,\ldots,n$$

La valeur λ_2 associée, supérieure ou égale à $\lambda_{1.1}$, est déterminée par l'algorithme suivant (cas particulier de l'algorithme G lorsque $u_j = 1$ $\forall j$) :

1 $\lambda_2 \leftarrow \lambda_{1.1}$; $j \leftarrow i$;

2 Si $j < n$ alors $j \leftarrow j + 1$ sinon stop

3 Si $a_j \leq b$ alors $\lambda_2 \leftarrow \lambda_2 + c_j$;
$b \leftarrow b - a_j$;
si $b = 0$ alors stop

aller en 2

Il est à noter que $\lambda_2 = v_2(B) = v_2(B \mid x_i = 0)$.

H_3 - Guignard et Spielberg (1972) [K24]

Après le calcul de $v_2(B) = \lambda_2$, les auteurs prennent l'option de fixer la variable de base à 1 pour déterminer λ_3 par l'algorithme suivant :

1 $\lambda_3 \leftarrow \max \{\lambda_2, v_2(B \mid x_i = 1)\}$; $k \leftarrow i - 1$; $X_0 \equiv \emptyset$

2 $\lambda \leftarrow \max_{1 \leq j \leq k} v_2(B \mid x_\ell = 0 \ \forall \ell \in X_0 \cup \{j\} \ ; \ x_i = 1)$

3 Si $\lambda > \lambda_3$ alors $\lambda_3 \leftarrow \lambda$; stop
sinon $X_0 \equiv X_0 \cup \{k\}$; $k \leftarrow k - 1$;
Si $k > 1$ alors aller en 2 ;

4 Stop

Note

On posera $\lambda_3 = \max \{\lambda_2, v_2(B \mid x_i = 1)\}$.

□

H_4 - Ingargiola et Korsh (1973) [K28]

L'algorithme de réduction d'Ingargiola et Korsh contient implicitement la détermination de

$$\lambda_4 = \max \{v_{1.1}(B \mid x_i = 0),\ v_{1.1}(B \mid x_i = 1);$$

$$\max_{\substack{j=1,\ldots,n \\ j\neq i}} \{v_{1.1}(B \mid x_j = \varepsilon) \mid \varepsilon = 0 \text{ si } j < i,\ 1 \text{ sinon}\}\}$$

H_5 - Hansen (1974) [K26]

Lorsque $\sum_{j=1}^{n} a_j x_j^2 < b$, la valeur obtenue par H_2 peut être améliorée par l'algorithme suivant :

---

1 Ecart $\leftarrow b - \sum_{j=1}^{n} a_j x_j^2$; $\lambda_5 \leftarrow \lambda_2$;

2 Si il existe $k, \ell \in \{1,\ldots,n\}$ tels que
$(x_k^2 = 1,\ x_\ell^2 = 0)$ et $(c_k < c_\ell,\ a_k + \text{Ecart} \geq a_\ell)$
Alors Ecart $\leftarrow$ Ecart $- a_\ell + a_k$;
$\lambda_5 \leftarrow \lambda_5 + c_\ell - c_k$;
Si Ecart > 0 alors aller en 2

3 Stop

H_6 - Fayard, Plateau (1976) [K11]

La valeur obtenue par la méthode H_4 peut être améliorée sensiblement en prolongeant le calcul de $\lambda_{1.1}$ par celui de λ_2 pour obtenir :

$$\lambda_6 = \max \{v_2(B \mid x_i = 0),\ v_2(B \mid x_i = 1),$$

$$\max_{\substack{j=1,\ldots,n \\ j\neq i}} \{v_2(B \mid x_j = \varepsilon) \mid \varepsilon = 0 \text{ si } j < i,\ 1 \text{ sinon}\}\}$$

H_7 - Balas et Zemel (1977) [K2]

Etant donnée

$U = \{j_1, j_2, \ldots, j_{i-1}\}$, i et $L = \{j_{i+1}, \ldots, j_n\}$

la partition de $J = \{1, \ldots, n\}$ fournie par la méthode de résolution de $(\bar{B})$ du type N.K.R. ($\bar{x}_j = 1 \ \forall j \in U$, $0 \ \forall j \in L$, $(b - \sum_{j \in U} a_j)/a_i$ $j = i$) la procédure proposée par les auteurs est une généralisation de l'heuristique H_3 :

---

1 $\lambda_7 \leftarrow \sum_{j \in U} c_j$; $\underline{a} \leftarrow \min_{j \in J} a_j$

2 pour k = $j_1, j_2, \ldots, j_{i-1}$

trouver un minorant de $v(B \mid x_k = 0)$ comme suit :

$\lambda \leftarrow \sum_{j \in U \setminus \{k\}} c_j$; écart $\leftarrow b - \sum_{j \in U \setminus \{k\}} a_j$; $X_1 \equiv \emptyset$

pour h = $i, j_{i+1}, \ldots, j_n$

Si a_h = écart ou $a_h + \underline{a} \leq$ écart

alors $X_1 \equiv X_1 \cup \{h\}$

$\lambda \leftarrow \lambda + c_h$

écart $\leftarrow$ écart $- a_h$

Si écart $\neq 0$ et $\exists h \in (L \cup \{i\}) \setminus X_1$: $\underline{a}$-écart $\leq a_h <$ écart

alors $\lambda \leftarrow \lambda + c_h$;

Si $\lambda > \lambda_7$ alors $\lambda_7 \leftarrow \lambda$

3 Stop

III.1.2 - Algorithmes approchants polynômiaux (notés P.A.)

Les résultats du paragraphe III.1.1 montrent que l'algorithme glouton est au mieux 1/2-approchant dans le cas entier et n'est pas ε-approchant dans le cas borné (i.e. il est toujours possible de mettre en évidence un cas défavorable pour lequel la précision tend vers 1).

Il est alors nécessaire de mettre en jeu une *énumération* des so
lutions pour aboutir à une méthode approchante ; cette énumération ne pou
ra à l'évidence n'être que *partielle* pour assurer la croissance polynômi
des temps d'exécution avec la taille du problème ; cette exploration des
solutions sera accompagnée d'une ou plusieurs utilisations de l'algorith

Deux types d'algorithmes P.A. pour le problème du knapsack en
variables bivalentes sont présentés : le premier dû à Johnson [A7] s'ap
plique au cas particulier où $c_j = a_j \ \forall j$; le second élaboré par Sahni [
et repris par Chandra, Hirschberg et Wong [A1] est général.

III.1.2.1 - Algorithme P.A. pour le problème : (PB) max ax s.c. ax ≤ b ; x ∈ V

Cette méthode consiste à effectuer une partition des données :

le réel $T > 1$ étant donné, on pose
$\bar{A} = \{j \in \{1,\ldots,n\} \mid a_j > b/T\}$ et $\underline{A} = \{1,\ldots,n\} \backslash \bar{A}$

pour résoudre exactement le sous-problème $(PB \mid x_j = 0 \ \forall j \in \underline{A})$ - en énu
mérant au besoin toutes ses solutions réalisables - avant d'améliorer la
valeur obtenue par un algorithme glouton prenant alors en compte les don
de $\underline{A}$, non considérées dans la première phase de la méthode.

Cet algorithme - noté $PA_1(T)$, puisqu'il dépend du paramètre T
fournit une valeur approchée $v_A(B)$ et la solution $\underline{x}$ associée :

Algorithme $PA_1(T)$:

1 Déterminer $\bar{S} \subset \bar{A}$:

$$\sum_{j\in\bar{S}} a_j = \max_{S\subset\bar{A}} \{\sum_{j\in S} a_j \mid \sum_{j\in S} a_j \leq b\}$$

2 l'algorithme G appliqué au sous-problème

$$(PB \mid x_j = 1 \ \forall j \in \bar{S} \ ; \ x_j = 0 \ \forall j \in \bar{A}\backslash\bar{S})$$

fournit $\underline{S} \subset \underline{A}$: $\sum_{j\in\underline{S}} a_j \leq b - \sum_{j\in\bar{S}} a_j$

3 $v_A(PB) \leftarrow \sum_{j\in\bar{S}\cup\underline{S}} a_j$; $\underline{x}_j \leftarrow 1 \ \forall j \in \bar{S} \cup \underline{S}$, 0 sinon.

4 Stop

Théorème 14 (Johnson [A7])

L'algorithme $PA_1(T)$ est approchant polynômial $\forall\, T \geq 1/\varepsilon, \varepsilon \in]0,1[$ *donné ; sa complexité est* $O(n^k)$ *avec* $k = \lfloor T \rfloor$ *si* $T \notin \mathbb{N}$, $T - 1$ *sinon.*

Démonstration :

Soient $\varepsilon \in]0,1[$ et $T \geq 1/\varepsilon$.

(i) *l'algorithme $PA_1(T)$ est approchant :*

$\exists\, S^* \subset \{1,\ldots,n\}$: $v(PB) = \sum_{j\in S^*} a_j \leq b$.

Deux cas sont à envisager :

cas 1 : $\underline{S} = \underline{A}$:

en posant $\bar{S}^* = S^* \cap \bar{A}$ et $\underline{S}^* = S^* \cap \underline{A}$,

$$\text{alors } v(PB) = \sum_{j\in\bar{S}^*} a_j + \sum_{k\in\underline{S}^*} a_j.$$

par définition de $\bar{S}$, $\sum_{j\in\bar{S}^*} a_j \leq \sum_{j\in\bar{S}} a_j$

par hypothèse, $\sum_{j\in\underline{S}^*} a_j \leq \sum_{j\in\underline{S}} a_j$

$\Rightarrow v(PB) \leq v_A(PB)$ ....

.... $\Rightarrow v(PB) = v_A(PB)$.

cas 2 : $\underline{S} \subsetneq \underline{A}$:

$$\Rightarrow \exists\, h \in \underline{A} \quad \left.\begin{array}{l} v_A(PB) + a_h > b \\ a_h \leq b/T \end{array}\right\} \Rightarrow \left.\begin{array}{l} v_A(PB) > b - b/T \\ v(PB) \leq b \end{array}\right\} \ldots$$

$$\ldots\ \Rightarrow \left.\begin{array}{l} v_A(PB) > v(PB)\,(1 - 1/T) \\ T \geq 1/\varepsilon \end{array}\right\} \Rightarrow v_A(PB) > (1 - \varepsilon)\, v(PB)$$

d'où la conclusion.

(ii) *l'algorithme $PA_1(T)$ est de complexité polynômiale :*

* *complexité de l'étape 1* : étant donné $S \subset \bar{A}$,

$$\left.\begin{array}{l} \sum\limits_{j \in S} a_j \leq b \\ \forall j \in S \quad a_j > b/T \end{array}\right\} \Rightarrow |S|\,(b/T) < b \Rightarrow |S| < T \ldots$$

$\ldots\ \Rightarrow$ en posant $k = \lfloor T \rfloor$ si $T \notin \mathbb{N}$, $T - 1$ sinon,

$$|\{S \mid S \subset \bar{A} : \sum_{j \in S} a_j \leq b\}| \leq |\{S \mid S \subset \{1,\ldots,n\} : |S| \leq k\}| = \sum_{j=1}^{k} \binom{n}{j}$$

$$\leq \sum_{j=1}^{k} n^j = \frac{n^{k+1} - 1}{n - 1} - 1 \ldots.$$

$\ldots\ \Rightarrow$ l'étape 1 est de complexité $O(n^k)$.

* *la complexité de l'étape 2* est $O(n)$ puisque l'algorithme G est de complexité linéaire et que $|\underline{S}| \leq n$.

Ainsi, globalement, l'algorithme $PA_1(T)$ est de complexité $O(n^k)$

□

Notes

(i) T doit être choisi le plus petit possible pour minimiser la complexité de l'algorithme ; ainsi, en pratique, on choisira $T = 1/\varepsilon$ (ou $\lceil 1/\varepsilon \rceil$ si la condition $T \in \mathbb{N}$ est requise).

(ii) En fait, Johnson inclut en phase 2 de son algorithme un tri des données suivant l'ordre décroissant des a_j, $j \in \underline{A}$; il a été vu que ce tri est inutile pour démontrer que l'algorithme est approchant ; de plus, une méthode de tri étant de complexité $O(n \log_2 n)$, il est clair que l'algorithme $PA_1(T)$ est de complexité $O(n \log_2 n)$ si $\varepsilon > 1/2$ (i.e. $k = \lfloor 1/\varepsilon \rfloor < 2$), $O(n^k)$ sinon.

□

III.1.2.2 - Algorithme approchant pour (B) max cx s.c. $ax \leq b$; $x \in V$

Les données de (B) ayant été triées suivant l'ordre décroissant des c_j/a_j, l'énumération partielle de l'ensemble des solutions de (B) est réalisée par l'exploration de toutes les solutions de $(B \mid \Sigma x_j \leq k)$, $k \in \mathbb{N}$ étant donné ; toutes les combinaisons des coefficients de la contrainte telles que

$$\sum_{j \in S} a_j \leq b \quad \text{et} \quad |S| \leq k \tag{25}$$

sont donc énumérées et pour chaque S vérifiant (25), l'algorithme G est appliqué pour le sous-problème $(B \mid x_j = 1 \ \forall j \in S)$; la meilleure valeur de la fonction économique ainsi atteinte sera le résultat $v_A^k(B)$ de cet algorithme [noté $PA_2(k)$] :

Algorithme $PA_2(k)$

1 Ranger les données dans l'ordre décroissant des rapports c_j/a_j de sorte qu'après renumérotation des variables :

$$c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n$$

2 $v_A^k(B) \leftarrow \max \{ \sum_{j \in S} c_j + v_G(B \mid x_j = 1 \ \forall j \in S) \mid |S| \leq k \text{ et } \sum_{j \in S} a_j \leq b \}$

3 Stop

Théorème 15 (Sahni [A12])

L'algorithme $PA_2(k)$ est approchant polynômial en posant $k = \left\lceil \frac{1-\varepsilon}{\varepsilon} \right\rceil$ $\forall \varepsilon \in]0,1[$ *donné ; sa complexité est* $O(k\, n^{k+1})$

Démonstration

Soient $\varepsilon \in]0,1[$ et $k = \left\lceil \frac{1-\varepsilon}{\varepsilon} \right\rceil$

(i) *l'algorithme $PA_2(k)$ est approchant* :
$\exists\, S^* \subset \{1,\ldots,n\}$: $v(B) = \sum_{j \in S^*} c_j$ et $\sum_{j \in S^*} a_j \leq b$.
deux cas sont à envisager :

cas 1 : $|S^*| \leq k$: $v_A^k(B) = v(B)$, par construction.

cas 2 : $|S^*| \geq k+1$:
Soit $\bar{S} \subset S^*$: $|\bar{S}| = k$ et $\forall j \in \bar{S}$ $c_j \geq c_h$ $\forall j \in S^* \setminus \bar{S}$

$$\left.\begin{array}{l} v_A^k(B) \geq \sum_{j \in \bar{S}} c_j + v_G(B \mid x_j = 1 \ \forall j \in \bar{S}) \\ v(B) = \sum_{j \in \bar{S}} c_j + \sum_{j \in S^* \setminus \bar{S}} c_j \quad (27) \end{array}\right\} \ldots$$

$$\ldots \Rightarrow v(B) - v_A^k(B) \leq \sum_{j \in S^* \setminus \bar{S}} c_j - v_G(B \mid x_j = 1 \ \forall j \in \bar{S})$$

$$\leq v(\bar{B} \mid x_j = 1 \ \forall j \in \bar{S}) - v_G(B \mid x_j = 1 \ \forall j \in \bar{S}) \quad \left.\begin{array}{l} \\ \\ \text{corollaire du théorème 8} \end{array}\right\}$$

$$\ldots \Rightarrow v(B) - v_A^k(B) \leq c_\tau$$

avec τ indice de la variable de base de $(\bar{B} \mid x_j = 1 \ \forall j \in \bar{S})$.

$$(26),\ (28) \text{ et } \tau \notin \bar{S} \Rightarrow v(B) - v_A^k(B) \leq c^*_{k+1} \qquad (29)$$

($(k+1)^{\text{ème}}$ plus grand élément de la suite (c_j) $j \in S^*$)

$$(27) \text{ et } |S^* \backslash \bar{S}| \geq 1 \Rightarrow v(B) \geq (k+1)\, c^*_{k+1} \Big\} \ldots$$

$$(29)$$

$$\ldots\ldots \Rightarrow (v(B) - v_A^k(B))/v(B) \leq 1/(k+1) \Big\} \Rightarrow (v(B) - v_A^k(B))/v(B) \leq \varepsilon$$

$$k = \left\lceil \frac{1-\varepsilon}{\varepsilon} \right\rceil$$

(ii) *l'algorithme $PA_2(k)$ est de complexité polynômiale :*

Etant donné

$$\Omega = \{S \subset \{1,\ldots,n\} \mid |S| \leq k \ ; \ \sum_{j \in S} a_j \leq b\}$$

$$|\Omega| \leq |\{S \subset \{1,\ldots,n\} \mid |S| \leq k\}| = \sum_{j=1}^{k} \binom{n}{j} \Big\} \ldots$$

l'algorithme G (de complexité $O(n)$) est appliqué pour tout $S \in \Omega$

$\ldots\ldots \Rightarrow$ l'algorithme $PA_2(k)$ est de complexité $O(kn^{k+1})$ puisque d'une part, la complexité d'une méthode de tri est $O(n \log_2 n)$ et d'autre part, $\varepsilon < 1$ et $k \in \mathbb{N} \Rightarrow k \geq 1$.

□

Notes

(i) $\forall\, \varepsilon \in]0,1[\quad k = \left\lceil \frac{1-\varepsilon}{\varepsilon} \right\rceil \Rightarrow \frac{k}{k+1} \geq 1 - \varepsilon$; une suite de problèmes $(E_h)_{h \geq 3}$ à $k+2$ variables peut être construite de telle sorte que

$$\lim_{h \to +\infty} v_A^k(E_h)/v(E_h) = k/(k+1) :$$

Etant donné, $\forall h \geq 3$

$$(E_h) \quad \begin{cases} \max \ 2x_1 + hx_2 + \ldots + h\, x_{k+2} \\ \text{s.c.} \ \ x_1 + hx_2 + \ldots + hx_{k+2} \leq (k+1)h \\ \qquad x_j = 0 \text{ ou } 1 \ \ j = 1,\ldots,k+2 \end{cases}$$

$$v_A^k(E_h) = k\,h + 2 \text{ et } v(E_h) = (k+1)h$$

$$\Rightarrow v_A^k(E_h)/v_A(E_h) = \frac{k\,h+2}{(k+1)h} \xrightarrow[h\to+\infty]{} \frac{k}{k+1}$$

(ii) k étant donné, Sahni [A12] propose une autre mesure du degré de précision de son algorithme à partir des remarques suivantes :

$$(29) \Rightarrow v(B) - v_A^k(B) \leq \bar{c}_{k+1}$$

$((k+1)^{\text{ème}}$ plus grand élément de la suite (c_j) $j = 1,\ldots,n)$

$$v(B) \geq c_{max} = \max_{1\leq j\leq n} c_j$$

$$\ldots\ldots \Rightarrow (v(B) - v_A^k(B))/v(B) \leq \min\{\frac{1}{k+1}, \frac{\bar{c}_{k+1}}{c_{max}}\}.$$

□

III.1.3 - Algorithme approchants totalement polynômiaux (notés FPA)

Le *temps d'exécution* d'un algorithme *approchant polynômial* n'est une fonction polynômiale que de la taille du problème ; il est par contre *fonction exponentielle* de *l'inverse de la précision* (l'algorithme $PA_2^{(k)}$ de complexité $O(\lfloor 1/\varepsilon \rfloor\, n^{\lceil 1/\varepsilon \rceil})$ Cette croissance exponentielle disparait les algorithmes approchants totalement polynômiaux [voir Ibarra et Kim [et Lawler [A9]] pour lesquels le temps d'exécution est une fonction polynômiale à la fois de la taille du problème et de l'inverse de la précisio

Deux principes sont à la base de tels algorithmes : la résolutio par la programmation dynamique d'un problème "normalisé", et la partitio des données. Il est à noter que le premier principe est suffisant pour ab tir à un algorithme FPA ; le second vient encore améliorer les résultats théoriques.

III.1.3.1 - Résultats préliminaires (programmation dynamique)

L'énumération de l'ensemble des solutions d'un problème du knapsack peut être faite par la programmation dynamique de deux manières duales :

(i) d'une part, en définissant

$$f_k(y) = \max \{ \sum_{j=1}^{k} c_j x_j \mid \sum_{j=1}^{k} a_j x_j \leq y \; ; \; x_j = 0 \text{ ou } 1 \; j = 1,\ldots,k \}$$

pour $k \in \{1,2,\ldots,n\}$ et $y \in \{1,\ldots,b\}$

dans le but de calculer de façon récursive :

$$f_1(y) = \begin{cases} 0 & \text{si } y < a_1 \\ c_1 & \text{sinon} \end{cases} \qquad y = 1,\ldots,b$$

$$f_k(y) = \begin{cases} f_{k-1}(y) & \text{si } y < a_k \\ \max \{f_{k-1}(y) \; ; \; c_k + f_{k-1}(y - a_k)\} & \text{sinon} \end{cases}$$

pour $k = 2,\ldots,n$ et $y = 1,\ldots,b$

la valeur optimale de (B) étant à l'évidence $f_n(b)$; un tel algorithme est de complexité $O(nb)$.

(ii) d'autre part, en définissant

$$g_k(z) = \min \{ \sum_{j=1}^{k} a_j x_j \mid \sum_{j=1}^{k} c_j x_j \geq z \; ; \; x_j = 0 \text{ ou } 1 \; j = 1,\ldots,k \}$$

pour $k \in \{1,2,\ldots,n\}$ et $z \in \{1,\ldots, \sum_{j=1}^{n} c_j\}$

afin d'obtenir récursivement

$$g_1(z) = \begin{cases} +\infty & \text{si } z > c_1 \\ a_1 & \text{sinon} \end{cases} \qquad z = 1, 2,\ldots,z^*$$

$$g_k(z) = \begin{cases} g_{k-1}(z) & \text{si } z > c_k \\ \min \{g_{k-1}(z) \; ; \; a_k + g_{k-1}(z - c_k)\} & \text{sinon} \end{cases}$$

pour $k = 2,\ldots,n$ et $z = 1,\ldots,z^*$

z^*, valeur optimale de (B) - obtenue ainsi par un algorithme de complexité $O(n\ v(B))$ - est donnée par :

Théorème 16 (Fisher [A3])

$$v(B) = z^* = \min \{z \in \mathbb{N} \mid g_n(z+1) > b\}$$

Démonstration

$$\forall z \in \mathbb{N} : g_n(z+1) > b \Rightarrow \{x \in V \mid ax \leq b \text{ et } cx \geq z+1\} = \emptyset$$
$$\Rightarrow v(B) \leq z,$$

$$\left.\begin{array}{l} \text{d'où } v(B) \leq z^* = \min \{z \in \mathbb{N} \mid g_n(z+1) > b\} \\ g_n(z^*) \leq b, \text{ par définition de } z^* \Rightarrow v(B) \geq z^* \end{array}\right\} \Rightarrow v(B) = z^*.$$

□

Une conséquence directe des points (i) et (ii) est donc le

Théorème 17

La programmation dynamique résoud le problème du knapsack en variables 0-1 par un algorithme de complexité $O(n \times \min (b, v(B)))$.

Notes

(i) La détermination du procédé le plus efficace [terme $\min (b, v(B))$] peut être fait en comparant b et $\sum_{j=1}^{n} c_j$ ou mieux $\lfloor v(\bar{B}) \rfloor$.

(ii) Cet algorithme est dit *pseudo-polynômial* (chapitre 1, par. II) ; mais si $\exists \alpha \in \mathbb{N} : \forall n\ c_j, a_j \leq \alpha\ j=1,\ldots n$, alors tout algorithme de programmation dynamique est polynômial puisque de complexité $O(n^2)$ [1a

constante multiplicative de n^2 étant un multiple de α].

(iii) Les complexités évoquées tout au long de l'exposé sont des complexités en temps de calcul ; une digression est faite à ce niveau pour inclure les résultats très récents de Magazine et Oguz [A11] sur la complexité en espace mémoire [dite spatiale] ; on supposera ici que $b \leq v(B)$:

La détermination de $v(B)$ par la programmation dynamique décrite ci-dessus, est réalisée par un algorithme de complexité spatiale $O(n+b)$ [n : entrée des données ; b : calcul par récurrence].

Par contre lorsqu'il s'agit de déterminer la solution optimale associée à $v(B)$, il faut connaître la solution attachée à chaque $y \in \{1,\ldots,b\}$; d'où une complexité spatiale $O(nb)$ (identique à la complexité en temps de calcul, dite temporelle).

Dans le cas de l'utilisation d'une calculatrice ne permettant pas un tel encombrement mémoire, Magazine et Oguz proposent un algorithme moins performant en temps d'exécution, puisque de complexité temporelle $O(n \log_2 n\ b)$, mais dont la complexité spatiale est $O(b)$; cette méthode est la suivante :

Remarque préliminaire : déterminer $k^* = \max \{k \mid x_k^* = 1\}$ parallèlement à $v(B)$ n'augmente pas la complexité temporelle de la programmation dynamique puisqu'il suffit de déterminer

$$\forall k \in \{1,\ldots,n\}\ \forall y \in \{0,\ldots,b\}\quad ind_k(y) = \begin{cases} k & \text{si } f_k(y) > f_{k-1}(y) \\ ind_{k-1}(y) & \text{sinon} \end{cases}$$

$$\text{avec } ind_1(y) = \begin{cases} 1 \text{ si } y \geq a_1 \\ 0 \text{ sinon} \end{cases}$$

pour poser $k^* = ind_n(b)$. Il est important de noter pour la suite que

$\forall y \in \{0,\ldots,b\}$, en notant $x^*(y)$ la solution optimale de

$$\max cx \text{ s.c. } ax \leq y\ ;\ x \in V$$

alors $ind_n(y) = \max \{k \mid x_k^*(y) = 1\}$.

Le principe est alors d'itérer la phase 2 du procédé suivant décrit dans sa première itération.

Phase 1 : la résolution de (B) par la programmation dynamique fournit $v(B)$ et $k^* = \max \{k \mid x_k^* = 1\}$ (ce qui implique que $x_j^* = 0 \quad j = k+1,\ldots,n$).
$m = k^* - 1$ composantes de x^* sont à déterminer.

Phase 2 :
1 séparation de $(B \mid x_{k^*}^* = 1)$ en deux sous-problèmes de dimension identiques (à une unité près si m est impair) :
$(B^1) \equiv (B \mid x_{k^*}^* = 1 ; x_j = 0 \; j = \lfloor m/2 \rfloor + 1,\ldots,m)$
$(B^2) \equiv (B \mid x_{k^*}^* = 1 ; x_j = 0 \; j = 1,\ldots, \lfloor m/2 \rfloor)$
les résolutions de (B^1) et de (B^2) par la programmation dynamique fournissent respectivement :
$f^1_{\lfloor m/2 \rfloor}(y)$ et $ind^1_{\lfloor m/2 \rfloor}(y)$
$f^2_m(y)$ et $ind^2_m(y)$ $\qquad y = 0,\ldots,b - a_{k^*}$

tels que $v(B^1) = f^1_{\lfloor m/2 \rfloor}(b-a_{k^*})$ et $v(B^2) = f^2_m(b - a_{k^*})$.

2 détermination de
$y^* \in \{0,1,\ldots,b-a_{k^*}\} : f^1_{\lfloor m/2 \rfloor}(y^*) + f^2_m(b - a_{k^*} - y^*) = v(B)$
$\Rightarrow$ en posant $k_1^* = ind^1_{\lfloor m/2 \rfloor}(y^*)$ et $k_2^* = ind^2_m(b - a_{k^*} - y^*)$,
deux nouvelles valeurs de composantes de x^* sont déterminées,

$$x^*_{k_1^*} = x^*_{k_2^*} = 1.$$

En notant qu'à l'itération k de la phase 2 :

- 2^k valeurs de composantes de x^* sont déterminées

- le nombre d'opérations de chacun des 2^k problèmes résolus est borné par $h \dfrac{n-2^{k-1}}{2^k} b$, $h \in \mathbb{N}$,

au cours des $\log_2 n$ (au maximum) itérations de la phase 2, le nombre d'opérations est borné par

$$h \sum_{k=1}^{\log_2 n} 2^k \frac{n - 2^{k-1}}{2^k} b = h\,(n \log_2 n\, b - nb)$$

d'où la complexité temporelle globale $O(n \log_2 nb)$; la complexité spatiale est à l'évidence $O(n+b)$ [4 tableaux de dimension b sont suffisants en phase 2].

□

III.1.3.2 - Normalisation de (B)

Etant donné K un réel supérieur à 1, la résolution par la programmation dynamique (2ème procédé) du *problème normalisé*

$$(B_K) \quad \begin{cases} \max \sum_{j=1}^{n} \lfloor c_j/K \rfloor\, x_j \\ \text{s.c.} \sum_{j=1}^{n} a_j\, x_j \le b \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,n \end{cases}$$

est à l'évidence un algorithme de complexité $O(n\, v(B)/K)$. Le coût (en temps d'exécution) de résolution de (B_K) est donc K fois plus faible que celui de (B) ; un choix judicieux de K permet d'aboutir à un algorithme approchant totalement polynômial pour (B) en résolvant uniquement (B_K) par la programmation dynamique ; la valeur retenue par cet algorithme - noté $FPA_1(K)$ - est $\sum_{j\in S_K^*} c_j$ avec $S_K^* \subset \{1,\ldots,n\}$: $v(B_K) = \sum_{j\in S_K^*} \lfloor c_j/K \rfloor$ et $\sum_{j\in S_K^*} a_j \le b$:

Théorème 18 (Ibarra et Kim [A6], Lawler [A9])

$\forall \varepsilon \in]0,1[$ en posant $K = \frac{\varepsilon}{n} \max_{1\le j\le n} c_j$, la résolution par la programmation dynamique de (B_K) est un algorithme approchant totalement polynômial pour (B) ; sa complexité est $O(n^3/\varepsilon)$.

Démonstration

Soit $S_K^* \subset \{1,\ldots,n\}$: $v(B_K) = \sum_{j \in S_K^*} \lfloor c_j/K \rfloor$ et $\sum_{j \in S_K^*} a_j \leq b$

(la valeur de l'algorithme $FPA_1(K)$ est donc $v_A(B) = \sum_{j \in S_K^*} c_j$)

(i) *l'algorithme $FPA_1(K)$ est approchant :*

$$\forall j \in \{1,\ldots,n\} \quad c_j < K(\lfloor c_j/K \rfloor + 1) \ldots.$$

$$\ldots. \Rightarrow \left.\begin{array}{l} \forall S \subset \{1,\ldots,n\} \quad \sum_{j \in S} c_j < K \sum_{j \in S} \lfloor c_j/K \rfloor < K\,|S| \\ \exists\, S^* \subset \{1,\ldots,n\} : v(B) = \sum_{j \in S^*} c_j \\ v_A(B) \geq K\, v(B_K) \geq K \sum_{j \in S^*} \lfloor c_j/K \rfloor \end{array}\right\} \ldots.$$

$$\ldots. \Rightarrow \left.\begin{array}{l} v(B) - v_A(B) \leq K\,|S^*| \quad (30) \\ |S^*| \leq n \quad (31) \\ K = \frac{\varepsilon}{n} \max_{1 \leq j \leq n} c_j \end{array}\right\} \Rightarrow \left.\begin{array}{l} v(B) - v_A(B) \leq \varepsilon \max_{1 \leq j \leq n} c_j \\ \max_{1 \leq j \leq n} c_j \leq v(B) \end{array}\right\} \ldots$$

$$\ldots. \Rightarrow (v(B) - v_A(B)) / v(B) \leq \varepsilon$$

(ii) *l'algorithme $FPA_1(K)$ est totalement polynômial :*

$v(B) \leq \sum_{j=1}^{n} c_j \leq n \times \max_{1 \leq j \leq n} c_j \Rightarrow n\, v(B)/K \leq n^3/\varepsilon$, d'où la complexité $O(n^3/\varepsilon)$.

□

Le résultat suivant généralise et améliore celui du théorème 18.

Théorème 19

Si $\exists\, m \in \mathbb{R}_+$, $\underline{v}(B)$ et $\bar{v}(B)$: $\bar{v}(B) \leq m\, \underline{v}(B)$ (32)

alors en posant $\forall \varepsilon \in]0,1[\quad K = \varepsilon \underline{v}(B)/n$ *l'algorithme* $FPA_1(K)$ *est de complexité* $O(n^2/\varepsilon)$.

Démonstration

La preuve est calquée sur celle du théorème précédent :

(i) $\forall \underline{v}(B)$ (30),(31) et $K = \varepsilon \underline{v}(B)/n \Rightarrow v(B) - v_A(B) \leq \varepsilon \underline{v}(B) \leq \varepsilon v(B)$

(ii) $\forall \underline{v}(B) \; \forall \bar{v}(B) \; \frac{n\, v(B)}{K} = \frac{n^2}{\varepsilon} \frac{v(B)}{\underline{v}(B)} \leq \frac{n^2}{\varepsilon} \frac{\bar{v}(B)}{\underline{v}(B)}$ (32) $\Rightarrow \frac{n\, v(B)}{K} \leq m \frac{n^2}{\varepsilon} \; \forall n$

ce qui implique une complexité $O(n^2/\varepsilon)$ pour l'algorithme $FPA_1(K)$.

□

Corollaire (Lawler [A9])

$\forall \varepsilon \in]0,1[$ *en posant* $K = \frac{\varepsilon}{n} \max \{v_G^*(B), \max_{1 \leq j \leq n} c_j\}$, *l'algorithme* $FPA_1(K)$ *est complexité* $O(n^2/\varepsilon)$.

Démonstration

- rappel (voir partie II) : la résolution de $(\bar{B})$ par l'algorithme NKR fournit $U \subset \{1,\ldots,n\}$ et $i \notin U$:

$$v(\bar{B}) < \sum_{j \in U} c_j + c_i \leq 2 \max \{\sum_{j \in U} c_j, c_i\} \qquad (33)$$

- d'autre part, l'algorithme G^* fournit

$$v_G^*(B) \geq \sum_{j \in U} c_j \qquad (34)$$

- en posant $\underline{v}(B) = \max \{v_G^*(B), \max_{1 \leq j \leq n} c_j\}$

(33) et (34) $\Rightarrow \exists\ m = 2 : \bar{v}(B) = v(\bar{B}) < m\ \underline{v}(B)$

le théorème 19 permet alors de conclure, en notant que l'algorithme NKR est de complexité $O(n)$.

□

III.1.3.3 - Partition des données de (B)

La partition des données va aboutir à une complexité $O(n/\varepsilon^2)$ qui est meilleure que $O(n^2/\varepsilon)$ lorsque, pour $\varepsilon \in]0,1[$ fixé, n devient arbitrairement grand.

Etant donnés T et $K \in \mathbb{R}$, on pose

$$\bar{C} = \{j \in \{1,\ldots,n\} \mid c_j > T\} \ ; \ \underline{C} = \{1,\ldots,n\}\backslash\bar{C}$$

$$(B_{T,K}) \quad \begin{cases} \max \ \sum_{j\in\bar{C}} \lfloor c_j/K \rfloor\ x_j \\ \text{s.c.} \ \sum_{j\in\bar{C}} a_j \quad x_j \leq b \\ \qquad x_j = 0 \text{ ou } 1 \quad \forall j \in \bar{C} \end{cases}$$

$$S_{T,K} = \{S \subset \bar{C} \mid \sum_{j\in S} a_j \leq b\}$$

pour définir l'algorithme totalement polynômial suivant, dû à l'origine à Ibarra et Kim [A6], puis amélioré par Lawler [A9] :

Algorithme $FPA_2(K,T)$

1 résolution de $(B_{T,K})$

2 $v_A(B) \leftarrow \max_{S\subset S_{T,K}} \{ \sum_{j\in S} c_j + v_{1.1}(B \mid x_j = 1\ \forall j \in S\ ;\ x_j = 0\ \forall j \in$

Théorème 20

Etant donnés $\underline{v}(B)$ et $\bar{v}(B)$ tels que $\bar{v}(B) \leq m\ \underline{v}(B)$, $m \in \mathbb{R}_+$, en po

$$\forall\, \varepsilon \in \,]0,1[\quad K = \alpha_1\, \varepsilon^{k}\, \underline{v}(B) \text{ et } T = \alpha_2\, \varepsilon^{k-1}\, \underline{v}(B) \qquad (35)$$

avec $2 \leq k \in \mathbb{N}$; $\alpha_1, \alpha_2 \in]0,1[$ et $\alpha_1/\alpha_2+\alpha_2 \leq 1$

l'algorithme $FPA_2(K,T)$ *est approchant totalement polynômial ; sa complexité est* $O(n/\varepsilon^k)$

Démonstration

(i) *l'algorithme* $FPA_2(K,T)$ *est approchant :*

Soient $\varepsilon \in]0,1[$ et $S^* \subset \{1,\ldots,n\}$: $v(B) = \sum_{j\in S^*} c_j$.

En posant $\bar{S}^* = S^* \cap \bar{C}$ et $\underline{S}^* = S^* \cap \underline{C}$, il est clair que

$$\left.\begin{array}{l} v(B) = \sum_{j\in \bar{S}^*} c_j + \sum_{j\in \underline{S}^*} c_j \\ v_A(B) \geq K \sum_{j\in \bar{S}^*} \lfloor c_j/K \rfloor + v_{1.1}(B_{\underline{C}}, S^*) \\ \text{où } (B_{\underline{C}}, S^*) \equiv (B \mid x_j = 1\ \forall j \in \bar{S}^*\ ;\ x_j = 0\ \forall j \in \bar{C}\backslash\bar{S}^*) \end{array}\right\} \cdots$$

$$\cdots \rightarrow v(B) - v_A(B) \leq \Big(\sum_{j\in \bar{S}^*} c_j - K \sum_{j\in \bar{S}^*} \lfloor c_j/K \rfloor\Big) + \sum_{j\in \underline{S}^*} c_j - v_{1.1}(B_{\underline{C}}, S^*) \qquad (36)$$

or

$$\left.\begin{array}{l} Q_1 = \sum_{j\in \bar{S}^*} c_j - K \sum_{j\in \bar{S}^*} \lfloor c_j/K \rfloor < K\,|\bar{S}^*| \\ \qquad\qquad \forall j \in \bar{S}^* \qquad c_j > T \end{array}\right\} \rightarrow Q_1 < K\,\frac{v(B)}{T} \qquad (37)$$

De plus,

$$\left.\begin{array}{l} Q_2 = \sum_{j\in \underline{S}^*} c_j - v_{1.1}(B_{\underline{C}}, S^*) \leq v(B_{\underline{C}}, S^*) - v_{1.1}(B_{\underline{C}}, S^*) < c_{\underline{i}^*} \\ \text{où l'indice de la variable de base de } (\bar{B}_{\underline{C}}, S^*) : \underline{i}^* \in \underline{C} \end{array}\right\} \rightarrow Q_2 < T \qquad (38)$$

Donc

$$\left.\begin{array}{l}(36),\ (37)\ \text{et}\ (38) \Rightarrow v(B) - v_A(B) < K\,\dfrac{v(B)}{T} + T \\ K = \alpha_1\,\varepsilon^k\,\underline{v}(B),\ T = \alpha_2\,\varepsilon^{k-1}\,\underline{v}(B)\ \text{où}\ 2 \le k \in \mathbb{N}\end{array}\right\} \quad \ldots.$$

$$\left.\begin{array}{l}\ldots. \Rightarrow v(B) - v_A(B) < \left(\dfrac{\alpha_1}{\alpha_2}\,\varepsilon\, v(B) + \alpha_2\,\varepsilon^{k-1}\,\underline{v}(B)\right. \\ \qquad\qquad \varepsilon \in [0,1[\Rightarrow\ \varepsilon^{k-1} < \varepsilon \\ \qquad\qquad\qquad\qquad \underline{v}(B) \le v(B)\end{array}\right\} \quad \ldots.$$

$$\left.\begin{array}{l}\ldots. \Rightarrow v(B) - v_A(B) < \left(\dfrac{\alpha_1}{\alpha_2} + \alpha_2\right)\varepsilon\ v(B) \\ \dfrac{\alpha_1}{\alpha_2} + \alpha_2 \le 1\end{array}\right\} \Rightarrow v(B) - v_A(B) < \varepsilon\ \ v(B).$$

(ii) *l'algorithme* $FPA_2(K,T)$ *est totalement polynômial,* de complexité $O(n/\varepsilon^k)$ puisque

- La résolution de $(B_{T,K})$ par le procédé 2 de la programmation dynamique aboutit à un nombre d'opérations borné par $h\,\dfrac{n\,v(B)}{K} = T(n)$ avec $h \in \mathbb{N}$

$$\left.\begin{array}{l}\Rightarrow\ T(n) \le h\,\dfrac{n\,\bar{v}(B)}{K} \\ K = \alpha_2\,\varepsilon^k\,\underline{v}(B)\end{array}\right\} \longrightarrow \left.\begin{array}{l}T(n) \le \dfrac{h}{\alpha_2}\,\dfrac{\bar{v}(B)}{\underline{v}(B)}\,\dfrac{n}{\varepsilon^k}, \\ \exists\ m \in \mathbb{R}_+\ \bar{v}(B) \le m\,\underline{v}(B)\end{array}\right\} \Longrightarrow T(n) \le \frac{h\,m}{\alpha_2}\,\frac{n}{\varepsilon^k}$$

- Le nombre de solutions réalisables envisagées par ce procédé est borné par $v(B)/K$ donc par $\dfrac{m}{\alpha_2}\ \dfrac{1}{\varepsilon^k}$, et à chacune d'elles est associée le déroulement d'un algorithme glouton de complexité $O(n)$ (puisque $|\underline{C}| \le n$).

□

Corollaire (Lawler [A9])

Etant donnés

$$\underline{v}(B) = \max \{v_G^*(B), \max_{1\le j\le n} c_j\} \ , \ \bar{v}(B) = v(\bar{B})$$

alors en posant

$$\forall\ \varepsilon \in]0,1[\ K = \frac{\varepsilon^2}{4}\,\underline{v}(B) \text{ et } T = \frac{\varepsilon}{2}\,\underline{v}(B)$$

l'algorithme $FPA_2(K,T)$ *est de complexité* $O(n/\varepsilon^2)$.

Démonstration

Conséquence directe du théorème 20 dont les hypothèses sont vérifiées puisque

(i) (33) et (34) $\Rightarrow \underline{v}(B) \le 2\,\bar{v}(B)$

(ii) $k = 2$

(iii) $\alpha_1 = 1/4$ et $\alpha_2 = 1/2 \Rightarrow \alpha_1/\alpha_2 + \alpha_2 = 1$.

□

Note

Les choix des valeurs des paramètres k, α_1 et α_2 aboutissent à une minimisation des complexités théorique et expérimentale puisque $\varepsilon^2 > \varepsilon^k$ $\forall k > 2$, $\alpha_1 = 1/4 = \max_{\alpha_2 \in]0,1[} (\alpha_2 - \alpha_2^2)$ et enfin

$$\alpha_2 = 1/2 = \max \{\alpha_2 \mid 1/4\alpha_2 + \alpha_2 \le 1 \ ; \ \alpha_2 \in]0,1[\}.$$

□

III.2 - MAJORANTS DE v(B)

Il est possible d'aboutir à un majorant de v(B) meilleur (c'est-à de valeur plus faible) que le majorant classique $\lfloor v(\bar{B}) \rfloor$ en utilisant

- soit la relaxation du problème asymptotique de Gomory [G30] (voir Fréhel [G17] et Gondran [G31] pour les cas des variables entières et bornées)

- soit des relaxations des deux sous-problèmes de (B) obtenus par séparation sur la variable de base : $(B \mid x_i = 0)$ et $(B \mid x_i = 1)$

III.2.1 - Relaxation du problème asymptotique de Gomory

La résolution de

$$(AG) \quad \left[\begin{array}{l} v(\bar{B}) - \sum_{j\in U} d_j^0 + \max \sum_{j\in U} d_j^0 x_j - \sum_{j\in L} d_j^1 x_j - (c_i/a_i)\, x_s \\ \qquad \text{s.c.} \sum_{\substack{j=1 \\ j\neq i}}^{n} \alpha_j x_j + \alpha_s x_s \geq \beta \\ \qquad 0 \leq x_j \leq 1 \quad \forall j \in L \cup U \; ; \; x_s \geq 0 \end{array}\right.$$

où

$$\left[\begin{array}{l} d_j^0 = c_j - (c_i/a_i)a_j, \; d_j^1 = (c_i/a_i)\, a_j - c_j \\ \alpha_j = (a_j/a_i) - \lfloor a_j/a_i \rfloor \qquad \forall j \\ x_s = b - \sum_{j=1}^{n} a_j x_j \text{ et } \beta = (b/a_i) - \lfloor b/a_i \rfloor \end{array}\right.$$

aboutit au majorant $\lambda^1 = \lfloor v(AG) \rfloor$ de v(B) (voir[K11]).

En remarquant que $v(AG) = v(\bar{B})$ lorsque $\sum_{j\in U} \alpha_j \geq \beta$, la résolution de (AG) n'offre un intérêt que si $\sum_{j\in U} \alpha_j < \beta$, ce qui est équivalent à

$$\sum_{j\in U} \lfloor a_j/a_i \rfloor \geq \lfloor b/a_i \rfloor$$

(voir propositions 2.2 et 2.3 de [K38]).

Note

Des exemples sont présentés dans [K11].

□

III.2.2 - Relaxations de $(B \mid x_i = 0)$ et $(B \mid x_i = 1)$

a) Il suffit de résoudre les sous-problèmes $(\bar{B} \mid x_i = 0)$ et $(\bar{B} \mid x_i = 1)$ pour obtenir à l'évidence un minorant de $v(\bar{B})$ en posant :

$$\lambda^2 = \lfloor \max \{v(\bar{B} \mid x_i = 0),\ v(\bar{B} \mid x_i = 1)\} \rfloor$$

Notes

(i) $\forall j \in L \cup U \quad \max \{v(\bar{B} \mid x_j = 0),\ v(\bar{B} \mid x_j = 1)\} = v(\bar{B})$

(ii) $\lambda^2 \leq \lambda^1$ puisque la contrainte $0 \leq x_i \leq 1$ est relachée en $x_i \geq 0$ dans le problème (AG).

□

b) Un autre type de majorant de $v(B)$, supérieur à λ^2 (mais obtenu plus rapidement) est proposé dans [K42] :

en notant
$$\begin{cases} \bar{i} : c_{\bar{i}}/a_{\bar{i}} = \max \{c_j/a_j \mid j \in L\} \\ \underline{i} : c_{\underline{i}}/a_{\underline{i}} = \min \{c_j/a_j \mid j \in U\} \end{cases}$$

$$\lambda^3 = \sum_{j \in U} c_j + c_{\bar{i}}(b - \sum_{j \in U} a_j)/a_{\bar{i}}$$

$$= \sum_{j \in U} c_j + \max \{ \sum_{j \in L} c_j x_j \mid \sum_{j \in L} a_j x_j \leq b - \sum_{j \in U} a_j \ ;\ x_j \in \mathbb{N}\ \forall j \in L\}$$

$$\geq v(\bar{B} \mid x_j = 1\ \forall j \in U\ ;\ x_i = 0) = v(\bar{B} \mid x_i = 0)$$

$$\lambda^4 = \sum_{j\in U} c_j + c_i - \underline{c_i}\Big(\sum_{j\in U} a_j + a_i - b\Big)/\underline{a_i}$$

$$= \sum_{j\in U} c_j + c_i - \min\Big\{\sum_{j\in U} c_j x_j \;\Big|\; \sum_{j\in U} a_j x_j \geq \sum_{j\in U} a_j + a_i - b \;;\; x_j \; \forall j \in$$

$$\geq v(\bar{B} \mid x_j = 0 \;\forall j \in L \;;\; x_i = 1) = v(\bar{B} \mid x_i = 1)$$

$$\Rightarrow \lambda^5 = \lfloor \max(\lambda^3, \lambda^4) \rfloor \geq \lambda^2.$$

□

III.3 - MINORANT DE AX*

Etant donné $\underline{v}(B)$, la recherche d'une solution x de F(B) telle que $cx > \underline{v}(B)$ est à la base de la détermination d'un minorant de ax^* obtenu en résolvant :

$$(Y) \qquad \begin{cases} \min \; ax \\ \text{s.c. } cx \geq \underline{v}(B) + 1 \\ \qquad x \in [V] \end{cases}$$

pour poser $\underline{b} = \lceil v(Y) \rceil$.

La solution de (Y) se déduit du

Théorème 21

Etant donné $z \in \mathbb{N}$, *une solution optimale* $\tilde{x}$ *de*

$$\min ax \text{ s.c. } cx \geq z \;;\; x \in [V]$$

est définie par la détermination de $U' \subset J = \{1,\ldots,n\}$ *tel que*

$$\begin{cases} \sum_{j\in U'} c_j + c_q > z \geq \sum_{j\in U'} c_j \\ \forall j \in U' \; a_j/c_j \leq a_p/c_p \quad \forall p \notin U' \\ a_q/c_q = \min\{a_p/c_p \mid p \notin U'\} \end{cases}$$

comme suit :

$$\begin{cases} \tilde{x}_j = 1 & \forall j \in U' \\ \tilde{x}_q = (z - \sum_{j \in U'} c_j)/c_q & \\ \tilde{x}_j = 0 & \forall j \in L' = J \setminus (U' \cup \{q\}) \end{cases}$$

Démonstration

Evidente dans ce cas particulier de la programmation linéaire.

□

Corollaire

(i) $z = \underline{v}(B) + 1 \Rightarrow U' = U \; ; \; q = i \; ; \; L' = L$

(ii) $\underline{b} \in [\sum_{j \in U} a_j + 1, \; b]$

Démonstration

Voir théorème 2.2 de [K38].

□

Notes

(i) $\underline{b}$ sera d'autant plus proche de b que $\underline{v}(B)$ sera grand et a_i grand par rapport à c_i.

(ii) un minorant de ax^*, supérieur à $\underline{b}$ est obtenu par séparation de (Y) :

$$\lceil \min \{v(Y \mid x_i = 0), \; v(Y \mid x_i = 1)\} \rceil .$$

□

III.4 - ENCADREMENT DE ΣX_j

L'encadrement de la fonction économique et de la contrainte :

$$\begin{cases} \underline{v}(B) < cx \leq \bar{v}(B) & (39) \\ \underline{b} \quad \leq ax \leq b & (40) \end{cases}$$

permet la détermination d'un *majorant* (inférieur à 2^n) *du nombre de solutions de* (B) à envisager :

En posant (voir [G25])

$$\ell = \max(\ell_1, \ell_2) \text{ avec } \begin{cases} \ell_1 = \min \{k \mid \sum_{j=1}^{k} c_j > \underline{v}(B),\ c_1 \geq c_2 \geq \ldots \geq c_n\} \\ \ell_2 = \min \{k \mid \sum_{j=1}^{k} a_j \geq \underline{b},\quad a_1 \geq a_2 \geq \ldots \geq a_n\} \end{cases}$$

$$u = \min(u_1, u_2) \text{ avec } \begin{cases} u_1 = \max \{k \mid \sum_{j=1}^{k} c_j \leq \bar{v}(B),\ c_1 \leq c_2 \leq \ldots \leq c_n\} \\ u_2 = \max \{k \mid \sum_{j=1}^{k} a_j \leq b,\quad a_1 \leq a_2 \leq \ldots \leq a_n\} \end{cases}$$

on déduit que

$$\forall x \in F(B) = \{x \in V \mid x \text{ vérifie (39) et (40)}\}$$

$$\ell \leq \sum_{j=1}^{n} x_j \leq u$$

ce qui implique

$$|F(B)| \leq \sum_{k=\ell}^{u} \binom{n}{k} < \sum_{k=0}^{n} \binom{n}{k} = 2^n.$$

Une illustration numérique est fournie par les tableaux 2 et 3 :

- les neuf problèmes étudiés sont caractérisés dans le tableau 2

- le tableau 3 donne les valeurs de

$$\underline{v}(B) = v_2(B),\ \bar{v}(B) = \lambda^2,$$

$$\underline{b} = \lceil \min \{ax \mid cx \geq v_2(B) + 1 ;\ x \in [V]\} \rceil,$$

les bornes ℓ et u associées et les majorants correspondants de $|F(B)|$.

Il est à remarquer enfin, que les bornes ℓ_1, ℓ_2, u_1 et u_2 peuvent être déterminées par des algorithmes du type NKR, donc de complexité O(n) en moyenne. Ces résultats seront généralisés dans le cadre de la résolution du problème du knapsack "baudruche" ; les théorèmes et algorithmes associés seront donc détaillés au chapitre 4.

PROBLEME	VARIABLE	1	2	3	4	5	6	7	8	9	10	11	12	b
1 [K11]	c	101	20	79	29									11
	a	10	2	8	3									
	$[\bar{x}]$	1	0	0	0									
	x^*	0	0	1	1									
2 [K11]	c	8	15	15	5									17
	a	4	8	8	5									
	$[\bar{x}]$	1	1	0	0									
	x^*	0	1	1	0									
3 [K21]	c	6	8	10	11									20
	a	2	3	7	10									
	$[\bar{x}]$	1	1	1	0									
	x^*	0	1	1	1									
4 [K21]	c	6	8	10	11	9								20
	a	2	3	7	10	9								
	$[\bar{x}]$	1	1	1	0	0								
	x^*	0	1	1	1	0								
5 [K8,K9,K10]	c	52	17	40	793	813	768	724						837
	a	45	15	40	793	820	777	737						
	$[\bar{x}]$	1	1	1	0	0	0	0						
	x^*	1	1	0	0	0	1	0						
6 [A12]	c	11	21	31	33	43	53	55	65					110
	a	1	11	21	23	33	43	45	55					
	$[\bar{x}]$	1	1	1	1	1	0	0	0					
	x^*	1	1	1	0	1	1	0	0					
7 [K11]	c	12	42	65	35	55	457	57	45	2				650
	a	5	42	75	42	72	631	85	73	14				
	$[\bar{x}]$	1	1	1	1	1	0	0	0	0				
	x^*	1	0	0	0	0	1	0	0	1				
8 [G61]	c	3	1	1	5	10	15	17	15	18	20			90
	a	2	1	1	5	11	17	20	18	25	30			
	$[\bar{x}]$	1	1	1	1	1	1	1	1	0	0			
	x^*	1	1	1	1	0	1	1	1	1	0			
9 [K6]	c	14	13	15	10	8	13	10	10	5	7	7	6	51
	a	7	7	10	7	6	10	8	9	5	10	12	13	
	$[\bar{x}]$	1	1	1	1	1	1	0	0	0	0	0	0	
	x^*	1	1	1	1	0	1	1	0	0	0	0	0	

Tableau 2

Problème	n	$\underline{v}(B)$	$\bar{v}(B)$	$\underline{b}$	ℓ	u	$\lvert F(B)\rvert \leq$	2^n
1	4	101	110	11	2	2	6	16
2	4	28	32	16	2	3	10	16
3	4	24	32	13	3	3	4	16
4	5	24	32	13	3	3	10	32
5	7	833	839	825	2	4	91	128
6	8	139	164	90	3	5	182	256
7	9	313	471	381	1	3	129	512
8	10	67	77	77	4	8	837	1024
9	12	73	77	48	6	7	924	4096

Tableau 3

IV - RELAXATION LAGRANGIENNE

Un encadrement de v(B) et de ax^* étant connu, le problème (B) est donc supposé se formuler comme suit :

$$\begin{cases} \max \ cx \\ \text{s.c. } x \in F(B) \end{cases}$$

avec $F(B) = \{x \in V \mid \underline{v}(B) < cx \leq \bar{v}(B) \ ; \ \underline{b} \leq ax \leq b\}$

rappel : $\underline{v}(B) \geq \sum_{j \in U} c_j$, $\bar{v}(B) \leq \lfloor v(\bar{B}) \rfloor$ et $\underline{b} \geq \sum_{j \in U} a_j + 1$.

□

Les résultats suivants - nécessaires à l'exposé de l'algorithme de réduction de la partie V - concernent l'élimination de variables par utilisation des relaxations lagrangiennes.

IV.1 - RÉSULTATS GÉNÉRAUX

En considérant

- (B_*) $\begin{cases} \min \ cx \\ \text{s.c. } ax \geq \underline{b} \\ \qquad x \in V \end{cases}$
- les relaxations lagrangiennes respectives de (B) et de (B_*) définies $\forall u \in \mathbb{R}_+$:

  $(RB(u))$ $\max_x cx + u(b - ax)$ s.c. $x \in V$

  $(RB_*(u))$ $\min_x cx + u(\underline{b} - ax)$ s.c. $x \in V$

- les duaux respectifs de (B) et de (B_*)

  (D) $\min v(RB(u))$ s.c. $u \geq 0$

  (D_*) $\max v(RB_*(u))$ s.c. $u \geq 0$

les résultats des théorèmes 1 et 2 (§ I.1, Chapitre 1) aboutissent à la chaîne suivante d'inégalités :

Théorème 22

En notant $\bar{u}$(resp. u_*) *le multiplicateur associé à* $ax \leq b$ (resp. $ax \geq \underline{b}$) *à l'optimum de* $(\bar{B})$ (resp. $(\bar{B}_*)$) :

$$\forall u \in \mathbb{R}_+$$
$$v(RB_*(u)) \leq v(D_*) = v(\bar{B}_*) = v(RB(u_*)) \leq v(B_*) \leq \underline{v}(B) \leq v(B)$$
$$v(B) \leq \bar{v}(B) \leq v(\bar{B}) = v(RB(\bar{u})) = v(D) \leq v(RB(u))$$

qui implique, en conjonction avec le corollaire du théorème 5 (§ I.3, chapitre 1) :

Théorème 23

Etant donnés $\varepsilon \in \{0,1\}$, $k \in \{1,\ldots,n\}$ et $u \in \mathbb{R}_+$

$$\left.\begin{array}{l} \lfloor\, v(RB(u) \mid x_k = \varepsilon) \,\rfloor \leq \underline{v}(B) \\ \text{ou} \\ \lceil\, v(RB_*(u) \mid x_k = \varepsilon) \,\rceil > \bar{v}(B) \end{array}\right\} \Longrightarrow x_k \leftarrow 1 - \varepsilon$$

Ces critères d'élimination définitive de variables utilisant les relaxations lagrangiennes (RB(u)), dites *majorantes*, et $(RB_*(u))$, dites *minorantes*, se définissent plus précisément encore en introduisant la notion de *pénalité* :

Définition 14 :

Une pénalité est une borne inférieure de l'augmentation (resp. la diminution) apportée à la valeur d'une relaxation d'un problème en variables entières de minimisation (resp. maximisation) lorsqu'une contrainte supplémentaire est imposée.

En effet, étant donné $u \in \mathbb{R}_+$, en posant

$J^+(u) = \{j \in \{1,\ldots,n\} \mid c_j - ua_j > 0\}$ et $J^-(u) = \{j \in \{1,\ldots,n\} \mid c_j - ua_j$

ce qui implique que

$$v(RB_*(u)) = ub + \sum_{j \in J^-(u)} (c_j - ua_j) \text{ et } v(RB(u)) = ub + \sum_{j \in J^+(u)} (c_j - ua_j)$$

et en définissant les pénalités $d_j^{\varepsilon}(u)$ et $p_j^{\varepsilon}(u)$ relatives à la contrainte $x_j = \varepsilon$ ($j \in \{1,\ldots,n\}$ et $\varepsilon \in \{0,1\}$) et associées respectivement aux problèmes (B) et (B_*) :

$$d_j^0(u) = p_j^1(u) = c_j - ua_j \quad \forall j \in J^+(u)$$

$$d_j^1(u) = p_j^0(u) = ua_j - c_j \quad \forall j \in J^-(u)$$

les résultats du théorème 23 deviennent :

Corollaire :

Etant donnés $u \in \mathbb{R}_+$

$$\left.\begin{array}{l} \text{(i) } \exists\, k \in J^+(u) : \lfloor\, v(RB(u)) - d_k^0(u) \,\rfloor \le \underline{v}(B) \\ \underline{ou} \\ \exists\, k \in J^-(u) : \lceil\, v(RB_*(u)) + p_k^0(u) \,\rceil > \bar{v}(B) \end{array}\right\} \Longrightarrow x_k \leftarrow 1$$

$$\left.\begin{array}{l} \text{(ii) } \exists\, k \in J^-(u) : \lfloor\, v(RB(u)) - d_k^1(u) \,\rfloor \le \underline{v}(B) \\ \underline{ou} \\ \exists\, k \in J^+(u) : \lceil\, v(RB_*(u)) + p_k^1(u) \,\rceil > \bar{v}(B) \end{array}\right\} \Longrightarrow x_k \leftarrow 0$$

Démonstration : Voir théorème 3.1. de [K11].

□

IV.2 - RELAXATIONS LAGRANGIENNES PRIVILÉGIÉES

Parmi l'infinité des relaxations lagrangiennes de (B) et (B_*), seule la connaissance de 2 n + 4 d'entre elles, dites *privilégiées*, est suffisante pour appliquer ces tests d'élimination de variables ; ce sont celles associées aux multiplicateurs suivants :

u_0 : nombre arbitrairement grand
(pour privilégier $\underline{b} \le ax \le b$)

LILLE I

$u_\ell = c_\ell/a_\ell \qquad \ell = 1,\ldots,n$

[supposés tels que $c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n$ pour simplifier la suite de cet exposé]

$u_{n+1} = 0$ (pour privilégier $\underline{v}(B) < cx \leq \bar{v}(B)$.

Cette assertion découle du théorème suivant (le résultat relatif aux relaxations lagrangiennes majorantes a également été établi dans [K25]) :

Théorème 24

Etant donnés $\varepsilon \in \{0,1\}$, $k \in \{1,\ldots,n\}$, $u \in \mathbb{R}_+$ et $\ell \in \{0,\ldots,n+1\}$: $u \in [u_\ell, u_{\ell+1}]$, $\exists\ \ell_1, \ell_2, \ell_3$ et $\ell_4 \in \{\ell, \ell+1\}$:

$$v\,(RB(u)) - d_k^o(u) \geq v(RB(u_{\ell_1})) - d_k^o(u_{\ell_1})$$

$$v\,(RB_*(u)) + p_k^o(u) \leq v(RB_*(u_{\ell_2})) + p_k^o(u_{\ell_2})$$

$$v\,(RB(u)) - d_k^1(u) \geq v(RB(u_{\ell_3})) - d_k^1(u_{\ell_3})$$

$$v\,(RB_*(u)) + p_k^1(u) \leq v(RB_*(u_{\ell_4})) + p_k^1(u_{\ell_4})$$

Démonstration : Voir lemme 3.1. et théorème 3.2. de [K11] et [K38]

□

IV.3 - MEILLEURES RELAXATIONS LAGRANGIENNES

Parmi les 2 n + 4 relaxations lagrangiennes privilégiées, il en existe deux (l'une majorante et l'autre minorante) meilleures que les autres : ce sont celles qui permettent la plus grande réduction de (B), i.e. qui réalisent respectivement

- pour la majorante :

$$\max_{u=u_0,\ldots,u_{n+1}} \left| \{k \in \{1,\ldots,n\} \mid \lfloor v(RB(u) \mid x_k = \varepsilon) \rfloor \leq \underline{v}(B)\} \right| \quad (41)$$

- pour la minorante :

$$\max_{u=u_0,\ldots,u_{n+1}} \left| \{k \in \{1,\ldots,n\} \mid \lceil v(RB_*(u) \mid x_k = \varepsilon) \rceil > \bar{v}(B)\} \right| \quad (42)$$

Comme il est exclu du point de vue pratique de déterminer les deux meilleures relaxations lagrangiennes au sens de (41) et (42), à l'instar de la majorité des auteurs (qui n'envisagent que les relaxations majorantes), nous choisirons en première approximation :

$$v(RB(\bar{u})) = \min_{u \geq 0} v(RB(u)) \quad (43)$$

$$v(RB_*(u_*)) = \max_{u \geq 0} v(RB_*(u)) \quad (44)$$

où $\begin{cases} \bar{u} = u_i = c_i/a_i \\ u_* = u_r = c_r/a_r \end{cases}$ avec i et r indices des variables fractionnaires respectives de $(\bar{B})$ et de $(\bar{B}_*)$.

Ces dernières relations résultent de la théorie de la dualité, mais peuvent également se déduire des trois résultats suivants :

Propriété 1 (*lemme 3.1 de* [K11])

Etant donnés $u \in \mathbb{R}_+$ et $\ell \in \{0,1,\ldots, n+1\}$: $u \in [u_\ell, u_{\ell+1}]$

$\exists\, \ell_1, \ell_2 \in \{k, k+1\}$: $v(RB_*(u_{\ell_1})) \geq v(RB_*(u))$

$v(RB(u_{\ell_2})) \leq v(RB(u))$

Propriété 2 (*lemme 3.2. de* [K38])

(i) $v(RB(u_i)) \leq v(RB(u_{i-1})) \leq \ldots \leq v(RB(u_1)) < v(RB(u_0))$

(ii) $v(RB(u_i)) \leq v(RB(u_{i+1})) \leq \ldots \leq v(RB(u_n)) \leq v(RB(u_{n+1}))$

Propriété 3 (*lemme 3.3. de* [K38])

$$\text{(i)}\ v(RB_*(u_r)) \geq v(RB_*(u_{r-1})) \geq \ldots \geq v(RB_*(u_1)) \geq v(RB_*(u_0))$$
$$\text{(ii)}\ v(RB_*(u_r)) \geq v(RB_*(u_{r+1})) \geq \ldots \geq v(RB_*(u_n)) > v(RB_*(u_{n+1}))$$

Notes :

(i) les pénalités associées à $(RB(u_i))$ (resp.$(RB_*(u_r))$) ne sont rien d'autre que les coûts réduits à l'optimum de $(\bar{B})$ (resp.

(ii) on montre d'autre part que (théorème 3.4. de [K38])

$$v(RB(u_\ell)) = \max \{cx \mid ax \leq b,\ 0 \leq x_j \leq 1\ \forall j \neq k,\ (x_k)\}\ \ell = 1,\ldots$$
$$v(RB(u_{n+1})) = \sum_{j=1}^{n} c_j = \max \{cx \mid x \in V\}$$

et

$$v(RB_*(u_\ell)) = \min \{cx \mid ax \geq \underline{b}\ ;\ 0 \leq x_j \leq 1\ \forall j \neq k,\ (x_k)\}\ \ell = 1,\ldots$$
$$v(RB_*(u_{n+1})) = 0 = \min \{cx \mid x \in V\}.$$

□

Application numérique :

En posant $\varepsilon_k = \begin{cases} 0 \text{ si } k \in U \\ 1 \text{ si } k \in L \end{cases}$ et en remarquant que

$$v(\bar{P} \mid x_k = \varepsilon_k) = \min_{u \geq 0} v(RP(u) \mid x_k = \varepsilon_k) = v(RP(u_{i(k)} \mid x_k = \varepsilon_k))$$

où $x_{i(k)}$ est la variable de base de $(\bar{P} \mid x_k = \varepsilon_k)$

[$\Longrightarrow$ $i(k) \in U \cup \{i\}$ si $k \in L$ et $i(k) \in L \cup \{i\}$ si $k \in U$]

la détermination de

$$\underline{k} \in U : a_{\underline{k}} = \max_{j \in U} a_j \quad ; \quad \bar{k} \in L : a_{\bar{k}} = \max_{j \in L} a_j$$

et de

$\underline{\ell} = i(\underline{k})$ (resp. $\bar{\ell} = i(\bar{k})$) indice de la variable de base de $(\bar{P} \mid x_{\underline{k}} = 0)$ (resp. $(\bar{P} \mid x_{\bar{k}} = 1)$),

permet de conclure que

$$\left| \begin{array}{l} \forall k \in \{1,\dots,n\} \quad \forall u \notin [u_{\bar{\ell}}, u_{\underline{\ell}}] \\ \exists\, \ell \in \{\bar{\ell}, \bar{\ell}+1,\dots,\underline{\ell}\} : v(RP(u_\ell) \mid x_k = \varepsilon_k) \le v(RP(u) \mid x_k = \varepsilon_k) \end{array} \right.$$

cette remarque et la suivante

$$\left| \begin{array}{l} (43) \not\Longrightarrow v(RB(u_i) \mid x_k = \varepsilon_k) = \min\limits_{u \ge 0} v(RB(u) \mid x_k = \varepsilon_k) \\ (44) \not\Longrightarrow v(RB_*(u_r) \mid x_k = \varepsilon_k) = \max\limits_{u \ge 0} v(RB_*(u) \mid x_k = \varepsilon_k) \end{array} \right. \quad \forall k \in \{1,\dots,n\}$$

sont vérifiées dans les expériences et exemples numériques suivants :

** Expériences sur ordinateur :

Une étude sur l'utilisation des relaxations lagrangiennes majorantes avec des problèmes à coefficients tirés au hasard suivant la loi uniforme dans l'intervalle [0, 99] (sous la condition $\max\limits_{1 \le j \le n} a_j \le b < \sum\limits_{j=1}^{n} a_j$) donne les résultats suivants :

* dans le tableau 4 sont répertoriés, pour les problèmes de 50, 100, 500 et 1 000 variables les nombres moyens de variables éliminées en utilisant $(RB(u_k))$, k parcourant différents sous-ensembles D_k de $\{1,\dots,n\}$ (depuis $D_k = \{1,\dots,n\}$ jusqu'à $D_k = \{i\}$) (note : compte-tenu du temps de calcul prohibitif, les expériences relatives à $D_k = \{1,\dots,n\}$ ont été réduites à 3 pour 500 variables, sont inexistantes pour 1 000 variables).

* pour les 100 problèmes de 50 variables,

. la figure 3 représente l'histogramme relatif à la distribution de ces problèmes selon le nombre de variables éliminées en utilisant $(RB(u_k))$, $k \in \{1,\dots,n\}$ (l'amplitude élémentaire des classes étant de une

unité, l'ordonnée h_j d'une classe j d'amplitude nv_j et d'effectif np_j es définie par $h_j = np_j/nv_j$).

. dans le tableau 5 sont répertoriés les nombres $N_k > 0$ de probl pour lesquels la meilleure relaxation lagrangienne majorante est $(RB(u_{i+k}$ (N.B. : il existe deux problèmes pour lesquels $(RB(u_{i-1}))$ et $(RB(u_{i+1}))$ toutes deux les meilleures $\Longrightarrow \sum_k N_k > 100$).

** Exemples numériques :

. Exemple 1. (Tableau 6) : problème à 7 variables (n° 5 du tabl $[r = 6, i = 4, \underline{v}(B) = 836, \lfloor v(\bar{B}) \rfloor = 839$ et $\underline{b} = 829]$: les relaxations lag giennes minorantes ne jouent aucun rôle ; deux variables sont fixées à (utilisant $(RB(u_1))$ pour x_5, $(RB(u_4))$ pour x_7.

. Exemple 2. (Tableau 7) : problème à 3 variables $[r = 1, i = 3$ $\underline{v}(B) = 7, \lfloor v(\bar{B}) \rfloor = 8$ et $\underline{b} = 9]$: la solution optimale est déterminée dir tement en utilisant les relaxations lagrangiennes : $x_3^* = 1$ avec $(RB_*(u_1)$ $x_1^* = 1$ avec $(RB(u_2))$ ou $(RB(u_3))$, $x_2^* = 0$ avec $(RB_*(u_3))$ en tenant compte $x_1^* = 1$.

La figure 4 (resp. 5) représente pour l'exemple 1 (resp. 2) les graphes fonctions de $\mathbb{R}$ dans $\mathbb{R}$ définies par $u \longrightarrow v(RB(u))$ (en trait gras) et $u \longrightarrow v(RB_*(u))$ (en pointillés).

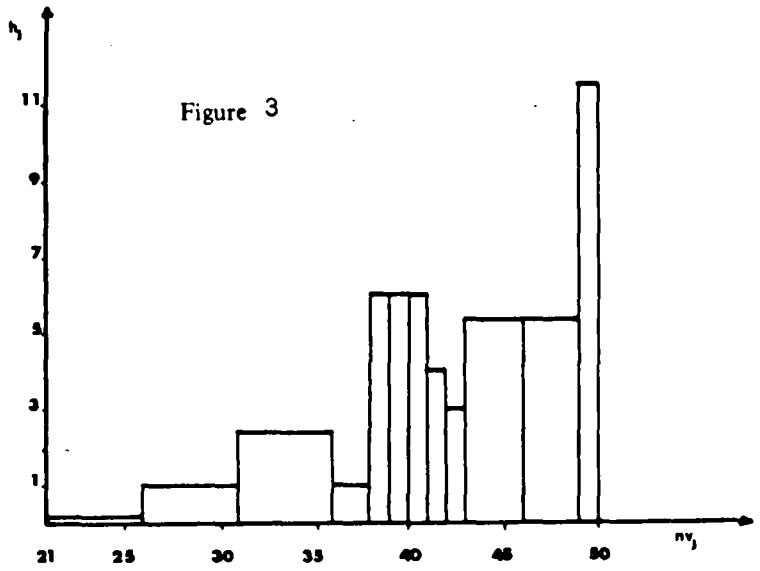


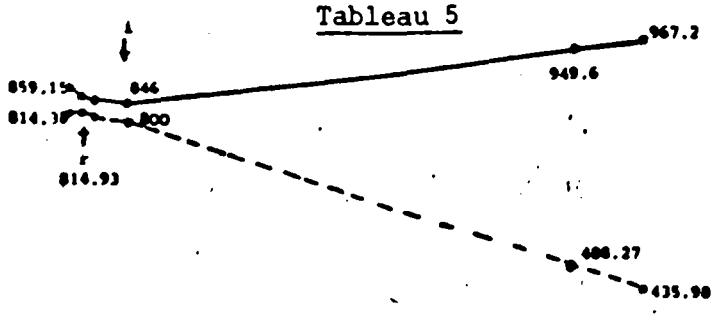
Figure 3

Nombre de		Nombre moyen de variables éliminées en utilisant					
Variables	Problèmes	$RB(u_k)$ k = 1,...,n	$RB(u_{i+k})$	$\|k\| \leq$	$RB(u_{i+k})$	$\|k\| \leq$	$RB(u_i)$
50	100	42.22	42.22	4	42.16	1	40.5
100	90	88.74	88.74	6	88.74	2	86.22
500	30	479	475	10	475	5	474.33
1 000	7		970.43	20	970.43	10	969.8

Tableau 4

k	-3	-1	0	+1
N_k	1	11	81	9

Tableau 5



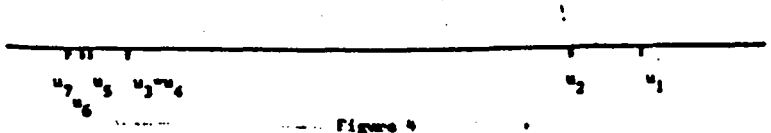


Figure 4

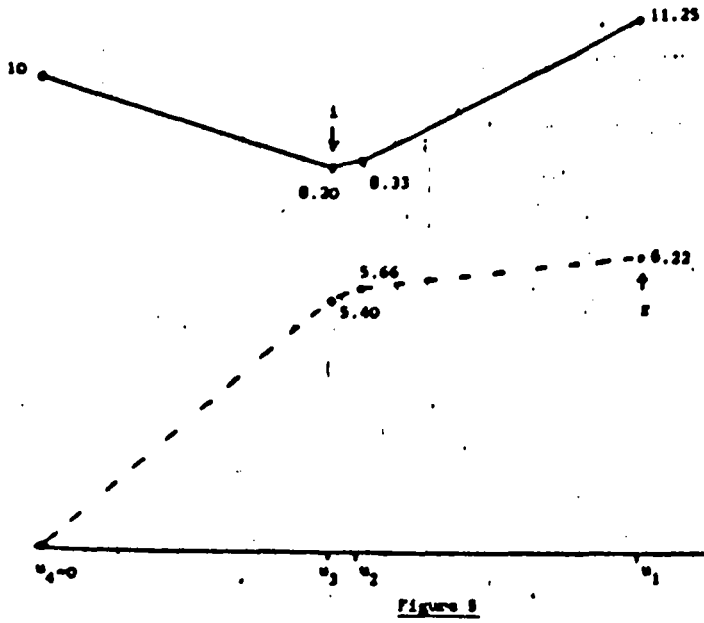


Figure 8

		c	52	17	40	793	813	768	724	b
		a	45	15	40	793	820	777	737	
k	u_k	$v(\bar{B}) - v(RB_*(u_k))$	$c_1-u_k a_1$	$c_2-u_k a_2$	$c_3-u_k a_3$	$c_4-u_k a_4$	$c_5-u_k a_5$	$c_6-u_k a_6$	$c_7-u_k a_7$	$v(RB(u_k))$
1	52/45	403.02	0	-0.33	-6.22	-123.35	-134.55	-129.87	-127.64	
							($x_5^*=0$)			
2	17/15	350.73	1	0	-5.33	-105.73	-116.33	-122.60	-111.27	
3,4	1	39	7	2	0	0	- 7	- 9	- 13	
									($x_7^*=0$)	
5	813/820	26.15	7.38	2.13	0.34	6.77	0	- 2.37	- 6.71	
6	768/777	24.07	7.52	2.17	0.46	9.18	2.50	0	- 4.46	
7	724/737	24.62	7.79	2.26	0.71	13.99	7.46	4.70	0	

Tableau 6

		c	5	2	3	b
		a	4	3	5	
k	u_k	$v(\bar{B})-v(RB_*(u_k))$	$c_1-u_k a_1$	$c_2-u_k a_2$	$c_3-u_k a_3$	$v(RB(u_k))$
1	5/4	1.75	0	-1.75	-3.25	
					($x_3^*=1$)	
2	2/3	2.33	2.33	0	-0.33	
			($x_1^*=1$)			
3	3/5	2.60	2.60	0.2	0	
			($x_1^*=1$)	($x_2^*=0$)		

Tableau 7

V - REDUCTION

La présentation d'un cadre commun pour les algorithmes de réduction de la taille de (B) (§ V.1) précède l'exposé d'une nouvelle méthode performante (§ V.3) mettant en jeu des relaxations, des relaxations lagrangiennes, et des relations binaires et de dominance entre variables (§ V.2).

V.1 - SYNTHÈSE

Les algorithmes de réduction du problème du knapsack entrent dans le cadre commun exposé au Chapitre 1 ; il se précise toutefois comme suit :

SCHEMA GENERAL DE REDUCTION DU KNAPSACK

1 Résolution de $(\bar{B})$
si $\bar{x} \in V$ alors $v(B) \leftarrow v(\bar{B})$; stop.
Soit x_i la variable de base de $(\bar{B})$

2 Utiliser une (ou plusieurs) heuristique (s) ou un algorithme approchant pour déterminer $\underline{v}(B)$

3 Déterminer $\bar{v}(B)$; si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$; stop.

4 Réduction de la taille de (B) : $X_0 \equiv X_1 \equiv \emptyset$ et $\varepsilon_k = \begin{cases} 0 \text{ si } k \in U \\ 1 \text{ si } k \in L \end{cases}$

4.1. Pour $k \in U \cup L$
Si $\bar{v}(B \mid x_k = \varepsilon_k) \leq \underline{v}(B)$ alors $x_k \leftarrow 1 - \varepsilon_k$;
$X_{1-\varepsilon_k} \equiv X_{1-\varepsilon_k} \cup \{k\}$

4.1.1. sinon si $\lambda_k = \underline{v}(B \mid x_k = \varepsilon_k) > \underline{v}(B)$
alors $\underline{v}(B) \leftarrow \lambda_k$
si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$; stop.

4.2. Pour $\varepsilon_i = 0$, puis 1
si $\bar{v}(B \mid x_i = \varepsilon) \leq \underline{v}(B)$ alors $x_i \leftarrow 1 - \varepsilon_i$;
$X_{1-\varepsilon_i} \equiv X_{1-\varepsilon_i} \cup \{i\}$

4.2.1. sinon si $\lambda_k = \underline{v}(B \mid x_i = \varepsilon_i) > \underline{v}(B)$

alors $\underline{v}(B) \leftarrow \lambda_k$;
si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$; stop.

Avant d'exposer de manière détaillée notre nouvel algorithme de réduction, la présentation du tableau 8 a pour but de mettre en évidence les particularités de chacun des 12 algorithmes de réduction connus (liste arrêtée au 30 Mai 1979) ; il indique successivement pour chacune des méthodes (considérées dans l'ordre chronologique)

- l'heuristique choisie
- la (les) majoration (s) envisagée (s) à l'étape 4 ;

Etant donnés X_0, X_1, $\varepsilon \in \{0,1\}$ et $k \in \{1,\ldots,n\}$, les trois types de majorations de $v(B \mid x_k = \varepsilon)$ rencontrés dans la littérature sont :

type 1 : $\lfloor v(RB(\bar{u}) \mid x_k = \varepsilon) \rfloor$ relaxation lagrangienne
type 2 : $\lfloor v(\bar{B} \mid x_k = \varepsilon) \rfloor$
type 3 : $\lfloor v(\bar{B} \mid x_k = \varepsilon \; ; x_j = 0 \; \forall j \in X_0 \; ; x_j = 1 \; \forall j \in X_1) \rfloor$ } relaxation

Notes :

(i) Toutes les méthodes (à l'exception de [K2] et [K13]) envisagent la résolution de la relaxation par l'algorithme C.K.R. (avec classement des données).

(ii) Dans le but de ne pas dédoubler le nombre de types, nous avons pris le parti de supposer que chaque algorithme utilise la partie entière dans le calcul de l'évaluation par excès (même dans le cas où l'auteur ne l'emploie pas).

(iii) Les étapes 4.1.1 et 4.2 n'ont un sens que dans le cadre de l'utilisation des relaxations (types 2 et 3).

(iv) L'étape 3 n'est envisagée que dans les méthodes [K31] et [K32] qui se démarquent des autres méthodes par le fait que le calcul de λ_6 contient implicitement les étapes 4.1.1 et 4.2.1 ainsi que les calculs de $v(\bar{B} \mid x_k = \varepsilon_k)$ avec $\varepsilon_k = 0$ si $k \in U \cup \{i\}$, 1 si $k \in L \cup \{i\}$.

(v) Dans nos méthodes [K11], [K12] les calculs des valeurs des relaxations tiennent compte du raisonnement suivant :
si $\exists\ p \in X_2$ tel que $a_p > b - \sum_{j \in X_1} a_j$ alors $x_p = 0\ \forall x \in F(B)$.

□

Méthode	$\underline{v}(B)$	Type de majoration de $v(B \mid x_k = \varepsilon)$
Fayard, Plateau [K8, K9, K10] (71, 72, 75)	λ_2	1
Ingargiola, Korsh [K28], (73)	λ_1	2
Dembo, Hammer [K6, K7] (74, 78)	λ_1	1
Hansen [K26] (74)	λ_5	1
Barr, Ross [K3] (75) Suhl [K41](78) Zoltners [K45] (78)	λ_2	1 et 3
Laurière [K31] (76) Martello, toth [K34] (77)	λ_6	2
Fayard, Plateau [K11, K12] (76, 77)	λ_2 puis λ_6	1 et 3
Nauss [K37] (76)	λ_3	1
Balas, Zemel [K2] (77)	λ_7	1 et 3

Tableau 8

V.2 - RÉSULTATS PRELIMINAIRES

La notion de dominance entre variables sera introduite dans le but de minimiser le volume des calculs de la phase effective de réduction évoquée précédemment. En outre, les relations binaires déterminées, en particulier, à partir des relaxations lagrangiennes, permettront d'affiner les évaluations par excès obtenues par relaxation.

V.2.1 - Relations de dominance

Dans le but de simplifier, étant donné $R = \bigcup_{j=1}^{n} \{c_j/a_j\}$, les éléments $U(J, R, k)$, $E(J, R, k)$ et $L(J, R, k)$ de la (R, k) partition seront respectivement remplacés par U_k, R_k et L_k.

D'autre part l'exposé de cette partie nécessite la donnée de la

Définition 15

Etant donnés $k, \ell \in J = \{1,\ldots,n\}$ *on dira que* x_k *domine* x_ℓ *si et seulement si* $\exists\ \varepsilon_1, \varepsilon_2 \in \{0,1\}$ $x_k \leftarrow \varepsilon_1 \Rightarrow x_\ell \leftarrow \varepsilon_2$; *on notera* $D_{\varepsilon_1}^{\varepsilon_2}$ *la relation d'ordre partiel associée i.e.*

$$x_k \, D_{\varepsilon_1}^{\varepsilon_2} \, x_\ell \quad \overset{\text{def.}}{\equiv} \quad (x_k \leftarrow \varepsilon_1 \Rightarrow x_\ell \leftarrow \varepsilon_2)$$

V.2.1.1 - Dominance entre variables d'indices $\in U \cup \{i\}$

Pour de telles variables, on va montrer que x_k domine x_ℓ lorsque les deux conditions suivantes sont réunies : $\ell \in U_k \cup E_k$ et $c_\ell \geq c_k$.

Lemme

Etant donnés $k \in U \cup \{i\}$ et $\ell \in U_k \cup E_k$
si $c_\ell \geq c_k$
alors (i) $d_\ell^0(u_i) \geq d_k^0(u_i)$

(ii) $v(\bar{B} \mid x_\ell = 0) \leq v(\bar{B} \mid x_k = 0)$

Démonstration : Voir lemme 1 de [K12].

□

Théorème 25

Etant donné $k \in U \cup \{i\}$
si $\exists \ell \in U_k \cup E_k$ *tel que* $c_\ell \geq c_k$

alors $x_k \; D_1^1 \; x_\ell$.

Démonstration : Voir théorème 1 de [K12].

□

Notes :

(i) Sous les hypothèses

$$\left[\begin{array}{l} c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n \qquad (T) \\ (\text{i.e. } U = \{1, 2, \ldots, i-1\} \text{ et } L = \{i+1, i+2, \ldots, n\}) \\ \quad \text{et } c_1 \geq c_2 \geq \ldots \geq c_{i-1} \geq c_i \end{array}\right.$$

le lemme du théorème 25 permet de conclure que :

$$\left|\begin{array}{l} d_1^0(u_i) \geq d_2^0(u_i) \geq \ldots \geq d_{i-1}^0(u_i) \geq d_i^0(u_i) = 0 \\ v(\bar{B} \mid x_1 = 0) \leq v(\bar{B} \mid x_2 = 0) \leq \ldots \leq v(\bar{B} \mid x_{i-1} = 0) \leq v(\bar{B} \mid x_i \end{array}\right.$$

c'est-à-dire $x_k \; D_1^1 \; x_\ell \; \forall \ell, k \in \{1, \ldots, i\} : \ell < k$.

(ii) Les résultats de ce lemme (i) se généralisent au cas des pénal
de toute relaxation lagrangienne privilégiée :
$p \in \{1, \ldots, n\}$ étant fixé on montre donc que :

$$\left|\begin{array}{l} \text{Pour } k \in U_p \cup E_p \text{ et } \ell \in U_k \cup E_k \\ \text{alors } c_\ell \geq c_k \Longrightarrow d_\ell^0(u_p) \geq d_k^0(u_p) \end{array}\right.$$

ce qui implique que :

$$\left|\begin{array}{l} \text{sous l'hypothèse (T) si } c_1 \geq c_2 \geq \ldots \geq c_p \\ \text{alors } d_1^0(u_p) \geq \ldots \geq d_{p-1}^0(u_p) \geq d_p^0(u_p) = 0. \end{array}\right.$$

□

V.2.1.2 - Dominance entre variables d'indices $\in L \cup \{i\}$:

Pour de telles variables, on va montrer que x_k domine x_ℓ lorsque les deux conditions suivantes sont réunies :

$\ell \in L_k \cup E_k$ et $a_k \leq a_\ell$

Lemme

Etant donnés $k \in L \cup \{i\}$ et $\ell \in L_k \cup E_k$
si $a_k \leq a_\ell$
alors (i) $d^1_k(u_i) \leq d^1_\ell(u_i)$
(ii) $v(\bar{B} \mid x_k = 1) \geq v(\bar{B} \mid x_\ell = 1)$.

Démonstration : Voir lemme 2 de [K12]

□

Théorème 26

Etant donné $k \in L \cup \{i\}$
si $\exists \ell \in L_k \cup E_k$ *tel que* $a_\ell \geq a_k$
alors $x_k \; D^O_O \; x_\ell$

Démonstration : Voir théorème 2 de [K12].

□

Notes :

(i) Sous l'hypothèse (T)

si $a_i \leq a_{i+1} \leq \ldots \leq a_{n-1} \leq a_n$
alors $x_k \; D^O_O \; x_\ell$ $\forall k, \ell \in \{i,\ldots,n\} : k < \ell$

puisque le lemme du théorème 26 conduit aux chaines d'inégalités

$0 = d^1_i(u_i) \leq d^1_{i+1}(u_i) \leq \ldots \leq d^1_{n-1}(u_i) \leq d^1_n(u_i)$

$v(\bar{B} \mid x_i=1) \geq v(\bar{B} \mid x_{i+1}=1) \geq \ldots \geq v(\bar{B} \mid x_{n-1}=1) \geq v(\bar{B} \mid x_n=1)$

(ii) Les résultats de ce lemme (i) se généralisent au cas des pénalités de toute relaxation lagrangienne privilégiée :

$p \in \{1,\ldots,n\}$ étant fixé :

Pour $k \in L_p \cup E_p$ et $\ell \in L_k \cup E_k$
alors $a_k \leq a_\ell \Longrightarrow d_k^1(u_p) \leq d_\ell^1(u_p)$

ce qui implique :

si $a_p \leq a_{p+1} \leq \ldots \leq a_n$
alors $0 = d_p^1(u_p) \leq d_{p+1}^1(u_p) \leq \ldots \leq d_n^1(u_p)$.

□

V.2.1.3. Dominance entre variables d'indices $\in \{1,\ldots,n\}$

Etant donnés $k \neq \ell \in \{1,\ldots,n\}$, si les deux conditions des théorèmes 25 et 26 sont réunies, c'est-à-dire si $c_k \geq c_\ell$ et $a_k \leq a_\ell$, alors $c_k/a_k > c_\ell/a_\ell$ (i.e. $\ell \in L_k \cup E_k$ et donc $k \in U_\ell \cup E_\ell$) ; les résultats des lemmes précédents peuvent ainsi être particularisés en considérant les valeurs de $(B \mid x_k = \varepsilon)$ et de $(B \mid x_\ell = \varepsilon)$, $\varepsilon \in \{0,1\}$, pour montrer que $x_\ell \; D_1^1 \; x_k$ et $x_k \; D_0^0 \; x_\ell$.

Lemme

Etant donnés $k, \ell \in \{1,\ldots,n\}$ *tels que* $k \neq \ell$
Si $a_k \leq a_\ell$ et $c_k \geq c_\ell$
alors $v(B \mid x_k = 0, x_\ell = 1) \leq v(B \mid x_k = 1, x_\ell = 0)$.

Démonstration

$$v(B \mid x_k = 0, x_\ell = 1) =$$

$$c_\ell + \max \left\{ \sum_{\substack{j=1 \\ j \neq k,\ell}}^{n} c_j x_j \;\middle|\; \sum_{\substack{j=1 \\ j \neq k,\ell}}^{n} a_j x_j \leq b - a_\ell,\ x_j = 0 \text{ ou } 1 \;\forall\, j \neq k,\ell \right\}$$

$$c_\ell \leq c_k$$

$$a_k \leq a_\ell \Rightarrow b - a_\ell \leq b - a_k$$

$$\cdots \Rightarrow v(B \mid x_k = 0,\ x_\ell = 1) \le c_k + \max \{ \sum_{\substack{j=1 \\ j\neq k,\ell}}^{n} c_j x_j \mid \sum_{\substack{j=1 \\ j\neq k,\ell}}^{n} a_j x_j \le b - a_k;$$

$$x_j = 0 \text{ ou } 1 \ \forall\ j \neq k,\ell\} = v(B \mid x_k = 1,\ x_\ell = 0).$$

□

Théorème 27

Etant donnés $k \neq \ell$; $k,\ell \in \{1,\ldots,n\}$
Si $a_k \le a_\ell$ et $c_k \ge c_\ell$
alors (i) $v(B \mid x_k = 0) \le v(B \mid x_\ell = 0)$
(ii) $v(B \mid x_k = 1) \ge v(B \mid x_\ell = 1)$
(iii) $x_\ell \ D_1^1 \ x_k$ et $x_k \ D_0^0 \ x_\ell$.

Démonstration

(i) et (ii) découlent du lemme précédent lorsqu'on a remarqué que, pour $\varepsilon \in \{0,1\}$ donné,
$v(B \mid x_k = \varepsilon) = \max \{v(B \mid x_k = \varepsilon,\ x_\ell = \varepsilon),\ v(B \mid x_k = \varepsilon,\ x_\ell = 1 - \varepsilon)\}$
et $v(B \mid x_\ell = \varepsilon) = \max \{v(B \mid x_\ell = \varepsilon,\ x_k = \varepsilon),\ v(B \mid x_\ell = \varepsilon,\ x_k = 1 - \varepsilon)\}$.
(iii) est une conséquence directe de (i) et (ii).

□

Corollaire

Si $c_1 \ge c_2 \ge \ldots \ge c_n$ *et* $a_1 \le a_2 \le \ldots \le a_n$
alors (i) $v(B \mid x_1 = 0) \le v(B \mid x_2 = 0) \le \ldots \le v(B \mid x_n = 0)$
(ii) $v(B \mid x_1 = 1) \ge v(B \mid x_2 = 1) \ge \ldots \ge v(B \mid x_n = 1)$
(iii) $x_\ell \ D_1^1 \ x_k$ *et* $x_k \ D_0^0 \ x_\ell \ \forall\ k < \ell$.

Démonstration : conséquence directe du théorème 27.

□

Notes :

Etant donnés k, $\ell \in \{1,\ldots,n\}$: $c_k/a_k \geq c_\ell/a_\ell$

i) $c_k \geq c_\ell$ et $a_k > a_\ell \Longrightarrow v(B \mid x_k = 0) \leq v(B \mid x_\ell = 0)$:

Considérons

$\max \{9 x_1 + 5 x_2 + 6 x_3 + 6 x_4 \mid 5 x_1 + 3 x_2 + 4 x_3 + 4 x_4 \leq 11,$

$x_j = 0$ ou 1, $j = 1,\ldots, 4\}$

$c_1 > c_2$ et $a_1 > a_2$, mais

$v(B \mid x_1 = 0) = 17 > v(B \mid x_2 = 0) = 15$

(par contre $v(\bar{B} \mid x_1 = 0) = 17 < v(\bar{B} \mid x_2 = 0) = 18$)

ii) $a_k \leq a_\ell$ et $c_k < c_\ell \Longrightarrow v(B \mid x_k = 1) \geq v(B \mid x_\ell = 1)$:

Considérons

$\max \{4 x_1 + 8 x_2 + 3 x_3 + 8 x_4 \mid 2 x_1 + 5 x_2 + 2 x_3 + 6 x_4 \leq 6,$

$x_j = 0$ ou 1, $j = 1,\ldots,4\}$

$a_3 < a_4$ et $c_3 < c_4$, mais

$v(B \mid x_3 = 1) = 7 < v(B \mid x_4 = 1) = 8$

(par contre $v(\bar{B} \mid x_3 = 1) = \frac{51}{5} > v(\bar{B} \mid x_4 = 1) = 8$).

□

V.2.2 - Relations binaires

On suppose être au cours de l'étape 4 d'un algorithme de réduction, donc connaître la partition X_0, X_1, X_2 de $\{1,\ldots,n\}$.

Etant donnés $\varepsilon \in \{0,1\}$ et $k \in X_2$, l'évaluation par excès $\bar{v}(B \mid x_k = \varepsilon)$ envisagée dans [K11] est obtenue par relaxation de

$$(B \mid x_k = \varepsilon;\ x_j = 0 \ \forall j \in X_0\ ;\ x_j = 1 \ \forall j \in X_1)$$

i.e. on résoud

$$\left[\begin{array}{l} \sum_{j\in X_1} c_j + c_k\,\varepsilon + \max \sum_{\substack{j\in X_2\\ j\neq k}} c_j\,x_j \\ \\ \text{s.c.} \sum_{\substack{j\in X_2\\ j\neq k}} a_j\,x_j \leq b - \sum_{j\in X_1} a_j - a_k, \\ \\ 0 \leq x_j \leq 1 \quad \forall j \in X_2 \setminus \{k\} \end{array}\right.$$

Cette évaluation, plus fine que celle utilisée dans [K2@], peut encore être affinée en considérant les relations binaires déduites des coupes fournies par

* Les relaxations lagrangiennes (i.e. $v(RB(u) \mid x_k = \varepsilon) > \underline{v}(B)$) :

Si $\exists \delta \in \{0,1\}$, $p \in X_2$, $u \in \mathbb{R}_+$: $\lfloor v(RB(u)) - d_k^{\varepsilon}(u) - d_p^{\varepsilon}(u) \rfloor \leq \underline{v}(B)$

alors $x_p \leftarrow 1 - \delta$ sous l'hypothèse $x_k = \varepsilon$

* La fonction économique : (i.e. $cx > \underline{v}(B)$)

si $\exists\, p \in X_2$ $\quad \sum_{\substack{j\in X_2\\ j\neq k,p}} c_j + c_k\,\varepsilon \leq \underline{v}(B) - \sum_{j\in X_1} c_j$

alors $x_p \leftarrow 1$ sous l'hypothèse $x_k = \varepsilon$

* La contrainte (i.e. $\underline{b} \leq ax \leq b$)

- Si $\exists\, p \in X_2$ $\quad a_p > b - \sum_{j\in X_1} a_j - a_k\,\varepsilon$

alors $x_p \leftarrow 0$ sous l'hypothèse $x_k = \varepsilon$

- Si $\exists\, p \in X_2$ $\quad \sum_{\substack{j\in X_2\\ j\neq k,p}} a_j + a_k\,\varepsilon < \underline{b} - \sum_{j\in X_1} a_j$

alors $x_p \leftarrow 1$ sous l'hypothèse $x_k = \varepsilon$

Ainsi en posant, $\delta \in \{0,1\}$ donné,

$X_\delta(x_k = \varepsilon) = \{p \in X_2 \mid x_p \leftarrow \delta$ sous l'hypothèse $x_k = \varepsilon\} \subset X_2$

l'évaluation par excès obtenue par relaxation de

$(B \mid x_k = \varepsilon\ ;\ x_j = 0 \quad \forall\, j \in X_o\ ;\ x_j = 1 \quad \forall\, j \in X_1\ ;$

$x_j = \delta \quad \forall\, j \in X_\delta(x_k = \varepsilon),\ \delta \in \{0,1\})$

sera meilleure (c'est à dire plus faible en valeur) que celles envisagées dans [K28] et [K11].

V.3 - NOUVEL ALGORITHME DE RÉDUCTION

Le principe du nouvel algorithme proposé est d'aborder quatre fois la phase effective de réduction (étape 4) de l'algorithme général en considérant, d'itération en itération, des majorations de plus en plus faibles en valeur - par conséquent de complexité de plus en plus forte - appliquées sur des ensembles de variables décroissants par inclusion.

Cet algorithme, qui entre dans le cadre commun exposé au paragraphe V.1., se particularise par les points suivants :

- l'heuristique choisie pour les minorations de l'étape 2 et de [illegible] les itérations de l'étape 4 est celle due à Greenberg et Hegerich

- la borne supérieure de l'étape 4 est
  $\lfloor \max \{v(\bar{B} \mid x_i = 0),\ v(\bar{B} \mid x_i = 1)\} \rfloor$

- les quatre itérations de l'étape 4 - dont le volume des calculs est en outre sensiblement diminué par l'apport des relations de dominance - sont enchainées de la manière suivante :

Itération 1 : utilisation de la relaxation lagrangienne (type 1) - déroulement uniquement de la partie 4.1. de l'algorithme - en prenant en compte les relations de dominance entre variables (voir également Walukiewicz [K[illegible]] et Zaltners [K45]) .

Itération 2 : $\forall k \in X_2 = J\setminus(X_0 \cup X_1)$: utilisation de la relaxation (type 3) en tenant compte des relations binaires déduites de $\underline{b} \leq ax \leq b$. En outre mise en mémoire des valeurs $\bar{\lambda}_k = \bar{v}(B \mid x_k = \varepsilon_k)$ supérieures au $\underline{v}(B)$ courant afin d'introduire le point suivant en fin d'itération (si $\underline{v}(B)$ a augmenté au cours des étapes 4.1.1. et 4.2.1. de l'itération) :

Pour $k \in X_2$ si $\bar{\lambda}_k \leq \underline{v}(B)$ alors $x_k \leftarrow 1 - \varepsilon_k$

$$X_{1-\varepsilon_k} \equiv X_{1-\varepsilon_k} \cup \{k\}$$

Itération 3 : $\forall k \in X_2$, utilisation de la relaxation (type 3) en tenant compte des relations binaires déduites de $\underline{b} \leq ax \leq b$, de $\underline{v}(B) + 1 \leq cx$ et de $\underline{v}(B) < \lfloor v(RB(\bar{u}) \mid x_k = \varepsilon_k) \rfloor$.

Itération 4 : $\forall k \in X_2$, en posant $(P \mid x_k = \varepsilon_k)$ la relaxation de $(B \mid x_k = \varepsilon_k)$ définie à l'itération 3, et en notant i_k sont indice de base optimale, utilisation de $\lfloor \max \{v(P \mid x_k = \varepsilon_k, x_{i_k} = 0), v(P \mid x_k = \varepsilon_k, x_{i_k} = 1)\} \rfloor$.

Note :

La généralisation d'un algorithme de réduction au cas des variables bornées permet de resserrer le domaine du knapsack en augmentant (resp. diminuant) les valeurs des bornes inférieures (resp. supérieures) des variables (voir par exemple Ingargiola et Korsh [K29]).

Exemple :

Considérons le problème à 50 variables suivant :

Minimiser

$$67x_1 + 30x_2 + 48x_3 + 95x_4 + 5x_5 + 90x_6 + 78x_7 + 91x_8 + 62x_9 + 84x_{10} +$$
$$97x_{11} + 98x_{12} + 43x_{13} + 96x_{14} + 73x_{15} + 44x_{16} + 95x_{17} + 54x_{18} + 94x_{19} + 60x_{20} +$$
$$35x_{21} + 38x_{22} + 74x_{23} + 80x_{24} + 74x_{25} + 57x_{26} + 39x_{27} + 55x_{28} + 28x_{29} + 56x_{30} +$$
$$39x_{31} + 65x_{32} + 36x_{33} + 53x_{34} + 48x_{35} + 42x_{36} + 19x_{37} + 16x_{38} + 17x_{39} + 25x_{40} +$$
$$41x_{41} + 26x_{42} + 27x_{43} + 19x_{44} + 16x_{45} + 12x_{46} + 9x_{47} + 11x_{48} + 10x_{49} + 5x_{50}$$

sous la contrainte

$$4x_1 + 2x_2 + 5x_3 + 10x_4 + x_5 + 19x_6 + 17x_7 + 26x_8 + 20x_9 + 32x_{10}$$
$$37x_{11} + 41x_{12} + 24x_{13} + 57x_{14} + 49x_{15} + 31x_{16} + 75x_{17} + 43x_{18} + 81x_{19} + 54x_{20}$$
$$32x_{21} + 38x_{22} + 78x_{23} + 92x_{24} + 90x_{25} + 70x_{26} + 49x_{27} + 70x_{28} + 36x_{29} + 72x_{30}$$
$$54x_{31} + 95x_{32} + 53x_{33} + 87x_{34} + 79x_{35} + 73x_{36} + 38x_{37} + 33x_{38} + 36x_{39} + 53x_{40}$$
$$91x_{41} + 78x_{42} + 90x_{43} + 93x_{44} + 80x_{45} + 65x_{46} + 59x_{47} + 73x_{48} + 78x_{49} + 91x_{50}$$

$x_j = 0$ ou $1 \qquad j = 1,\ldots, 50$

dont les variables sont déjà triées de telle sorte que

$c_1/a_1 \geq c_2/a_2 \geq \ldots \geq c_n/a_n$

étape 1 :

$$\bar{x}_j = \begin{cases} 1 & j = 1,\ldots,31 \\ 68/95 & j = i = 32 \\ 0 & j = 33,\ldots,50 \end{cases}, \qquad u_i = 65/95, \ v(\bar{B}) = 2025 . 526$$

étape 2

$\underline{v}(B) = \underline{v}_2(B) = 2015$ (à noter que $v_1(B) = \sum_{j=1}^{i-1} c_j = 1979 = c\,[\bar{x}]$).

étape 3

$\bar{v}(B) = 2024$

étape 4

Itération 1 :

35 variables sont éliminées par la relaxation lagrangienne, les relations de dominance étant utilisées de la manière suivante :

a) pour les variables d'indices $i-1, i-2, \ldots, 1$:

0 $j \leftarrow i - 1$

1 Si $\lfloor v(\bar{B}) - (c_j - u_i a_j) \rfloor < \underline{v}(B)$ alors aller en 2
sinon $j \leftarrow j - 1$, aller en 1

2 $x_j \leftarrow 1, \quad p \leftarrow j - 1$

3 si $c_p \geq c_j$ alors $x_p \leftarrow 1$, $p \leftarrow p - 1$, aller en 3

4 si $\lfloor v(\bar{B}) - (c_p - u_i a_p) \rfloor < \underline{v}(B)$ alors $x_p \leftarrow 1$, $j \leftarrow p$

5 $p \leftarrow p - 1$, aller en 3

b) pour les variables d'indices $i+1, i+2, \ldots, n$:

0 $j \leftarrow i + 1$

1 si $\lfloor v(\bar{B}) - (u_i a_j - c_j) \rfloor < \underline{v}(B)$ alors aller en 2

sinon $j \leftarrow j + 1$, aller en 1

2 $x_j \leftarrow 0$, $p \leftarrow j + 1$

3 si $a_p \geq a_j$ alors $x_p \leftarrow 0$, $p \leftarrow p + 1$, aller en 3

4 si $\lfloor v(\bar{B}) - (u_i a_p - c_p) \rfloor < \underline{v}(B)$ alors $x_p \leftarrow 0$, $j \leftarrow p$

5 $p \leftarrow p + 1$, aller en 3

Ainsi, pour ce problème :

a) $j = 31, 30, \ldots, 1$

- $x_{31}, x_{30}, \ldots, x_{26}$ non éliminées
- $x_{25} \leftarrow 1$, $c_{25} = 74 \Rightarrow x_{24}, x_{23} \leftarrow 1$
- $x_{22} \leftarrow 1$, $c_{22} = 38$
- $x_{21} \leftarrow 1$, $c_{21} = 35 \Rightarrow x_{20}, x_{19}, \ldots, x_6 \leftarrow 1$
- x_5 non éliminée

$\Rightarrow x_4, x_3 \leftarrow 1$

- $x_2 \leftarrow 1 \quad c_2 = 30 \Rightarrow x_1 \leftarrow 1$

b) $j = 33, 34, \ldots, 50$

- $x_{33}, \ldots, x_{38}$ non éliminées
- $x_{39} \leftarrow 0$, $a_{39} = 53 \Rightarrow x_{40}, x_{41}, \ldots, x_{50} \leftarrow 0$

le gain total est de 31 multiplications, 31 soustractions pour une perte de 11 comparaisons.

itération 2 :

On considère le sous-problème à 15 variables d'indices $\in X_2 = \{5, 26, 27, \ldots$

Une nouvelle évaluation par défaut est trouvée $\underline{v}(B) = 2016$; une seule variable est éliminée :

$$\lfloor v(\bar{B} \mid x_{26} = 0) \rfloor = \lfloor v(\bar{B}) - d^{\circ}_{26}(u_i) \rfloor = 2016 = \underline{v}(B) \Rightarrow x_{26} \leftarrow 1$$

Itération 3 :

7 variables sont éliminées en considérant le sous-problème à 14 variables suivant

$$(P_1) \quad \left\lceil \begin{array}{l} 1757+\max\ 5x_5 + 39x_{27} + 55x_{28} + 28x_{29} + 56x_{30} + 39x_{31} + 65x_{32} + \\ \text{s.c.}\ 36x_{33} + 53x_{34} + 48x_{35} + 42x_{36} + 19x_{37} + 16x_{38} + 17x_{39} \\ \quad x_5 + 49x_{27} + 70x_{28} + 36x_{29} + 72x_{30} + 54x_{31} + 95x_{32} + \\ 53x_{33} + 87x_{34} + 79x_{35} + 73x_{36} + 38x_{77} + 33x_{38} + 36x_{39} \leq 350 \\ x_j = 0 \text{ ou } 1 \qquad j = 5, 27, 28, \ldots, 39 \end{array} \right.$$

dont les coûts réduits à l'optimum de $(\bar{P}_1)$ sont les suivants

j	5	27	28	29	30	31
$d^{\circ}_j(u_i)$	4.316	5.474	7.105	3.368	6.737	2.053

j	33	34	35	36	37	38	39
$d^{1}_j(u_i)$	.263	6.526	6.053	7.947	7	6.579	7.632

Le tableau 9 montre l'intérêt des relations binaires déduites de la relaxation lagrangienne pour les variables d'indices j = 34,..,39 qui sont ainsi éliminées à la valeur 0 (ces relations n'apportent pas d'amélioration par rapport à l'itération 2 pour j = 5, 27,.., 33).

On y notera

$(\tilde{P}_1 \mid x_k = 1)$ la relaxation de $(P_1 \mid x_k = 1,\ x_j = \delta\ \forall\ j \in X_\delta(x_k=1),\ \delta \in \{0,1\})$

k	$v(\bar{P}_1 \mid x_k = 1)$	$X_1(x_k = 1)$	$X_o(x_k = 1)$	$v(\tilde{P}_1 \mid x_k = 1)$
34	2018	5, 27,..., 31	35,..., 39	$-\infty$ (second membre= -19)
35	2019	5, 27,..., 30	34, 36,..., 39	1988
36	2017	5, 27,..., 31	34,35,37,38,39	$-\infty$ (second membre= - 5)
37	2018	5, 27,..., 31	34,35,36,38,39	1988
38	2018	5, 27,..., 31	34,..., 37, 39	1995
39	2017	5, 27,..., 31	34,..., 38	1996

Tableau 9

On aboutit alors à l'ensemble $X_2 = \{5, 27,..., 33\}$ avant de calculer $v(\bar{P}_1 \mid x_{32} = \varepsilon)$ $\varepsilon \in \{0,1\}$, où x_{32} est la variable de base optimale de (B) donc de (P_1) - il est inutile de calculer $v(\tilde{P}_1 \mid x_{32} = \varepsilon)$ puisque le coût réduit associé à x_{32} est nul - on trouve ainsi $v(\bar{P}_1 \mid x_{32} = 0) = -\infty$ puisque $\sum_{\substack{j \in X_2 \\ j \neq 32}} a_j < \underline{b} = 337$

Itération 4 :

Le sous problème considéré est

$$\begin{cases} 1822 + \max\ 5x_5 + 39x_{27} + 55x_{28} + 28x_{29} + 56x_{30} + 39x_{31} + 36x_{33} \\ \text{s.c. } 242 \leq x_5 + 49x_{27} + 70x_{28} + 36x_{29} + 72x_{30} + 54x_{31} + 53x_{33} \leq 255 \\ x_j = 0 \text{ ou } 1 \quad j = 5, 27,...,31, 33. \end{cases}$$

dont les coûts réduits à l'optimum de $(\bar{P}_2)$ sont les suivants (la nouvelle variable de base est $x_{31} \Rightarrow u_i = 39/54$) :

j	5	27	28	29	30
$d_j^o(u_i)$	4.278	3.611	4.444	2	4

et $d_{33}^1(u_i) = 2.278$

Les variables x_{27}, x_{28}, x_{29} et x_{30} sont éliminées à la valeur 1 en séparant chacun des problèmes $(P_2|x_k = 0)$ $k \in \{27,\ldots, 30\}$, sur la variable de base associée. A titre d'exemple, le schéma d'élimination de x_{27} est le suivant :

x_{33} est la variable de base optimale de $(P_2|x_{27} = 0)$

$v(\tilde{P}_2 \mid x_{27} = 0, x_{33} = 0) = 2005$ puisque $X_1(x_{27} = 0, x_{33} = 0) = \{5,28,30\}$

$v(\tilde{P}_2 \mid x_{27} = 0, x_{33} = 1) = 2002$ puisque $X_1(x_{27} = 0, x_{33} = 1) = \{5,28,29,30\}$

donc $\max\{v(\tilde{P}_2 \mid x_{27} = 0, x_{33} = 0), v(\tilde{P}_2 \mid x_{27} = 0, x_{33} = 1)\} = 2005$
$< \underline{v}(B) = 2016$

Finalement on aboutit au problème

$$(P_3)\begin{cases} 2000+\max\ 5x_5 + 39x_{31} + 36x_{33} \\ \text{s.c.}\ \ 15 \le x_5 + 54x_{31} + 53x_{33} \le 28 \end{cases}$$

dont le domaine est vide puisque

$$\begin{cases} x_5 + 54x_{31} + 53x_{33} \le 28 \ \Rightarrow\ x_{31} \text{ et } x_{33} \leftarrow 0 \\ x_5 + 54x_{31} + 53x_{33} \ge 15 \ \Rightarrow\ x_{31} \text{ ou } x_{33} \leftarrow 1 \end{cases}$$

La solution optimale est donc celle associée à $\underline{v}(B) = 2016$ trouvée à l'itération 2.

□

VI - ENUMERATION IMPLICITE

Le problème (B) étant supposé non résolu par l'algorithme de réduction préliminaire

i.e. $\bar{x} \notin V$, $\underline{v}(B) < \bar{v}(B)$ et $|X_2| \neq \emptyset$

le schéma choisi pour l'énumération exhaustive de l'ensemble des solutions du problème réduit - dont le nombre de variables $|X_2|$ sera encore noté n - est l'exploration lexicographique :

A partir de $x^1 \in V$, les $2^n - 1$ autres sommets du cube unité $x^j (j = 2,3,\ldots,2^n)$ son générés dans l'ordre suivant :

$$
\begin{aligned}
x^2 &: (1 - x_1^1,\ x_2^1,\ x_3^1,\ldots,\ x_n^1)\\
x^3 &: (x_1^1,\ 1 - x_2^1,\ x_3^1,\ldots,\ x_n^1)\\
x^4 &: (1 - x_1^1,\ 1 - x_2^1,\ x_3^1,\ldots,\ x_n^1)\\
x^5 &: (x_1^1,\ x_2^1,\ 1 - x_3^1,\ldots,\ x_n^1)\\
x^6 &: (1 - x_1^1,\ x_2^1,\ 1 - x_3^1,\ldots,\ x_n^1)\\
&\ \ \vdots\\
x^{2^n} &: (1 - x_1^1,\ 1 - x_2^1,\ 1 - x_3^1,\ldots,\ 1 - x_n^1)
\end{aligned}
$$

Notes

(i) : Le choix de $x^1 \in V$ implique la définition d'une hiérarchie entre les variables (voir § VII).

(ii) En choisissant une partition $\{I,\bar{I}\}$ de $\{1,\ldots,n\}$ (on pose $m = |I|$), ce schéma exploratoire peut n'être appliqué que pour le cube de $\mathbb{R}^{n-m}$ (associé au sous-ensemble de variables d'indices appartenant à $\bar{I}$) dans le but de résoudre pour chaque vecteur bivalent $x_{\bar{I}}$ de $\mathbb{R}^{n-m}$ le problème auxiliaire paramétré :

$$(PA) \quad \left[\begin{array}{l} c_{\bar{I}}\, x_{\bar{I}} + \max c_I\, x_I \\ \qquad \text{s.c. } a_I\, x_I \leq b - a_{\bar{I}}\, x_{\bar{I}} \\ \qquad\quad x_j = 0 \text{ ou } 1 \quad \forall j \in I \end{array}\right.$$

(iii) L'autre type de procédure d'exploration dite LIFØ(i.e. Last In First Out) sera décrite dans l'exemple numérique détaillé en fin de cette partie.

□

L'arborescence associée au schéma exploratoire des solutions est parcourue implicitement comme suit :

Etant donné un noeud de cette arborescence, c'est-à-dire une partition $\{X_0, X_1, X_2\}$ de $\{1,\ldots,n\}$

$$\text{avec} \quad \left[\begin{array}{l} X_0 = \{j \mid x_j = 0\} \\ X_1 = \{j \mid x_j = 1\} \end{array}\right.$$

les test successifs suivants sont appliqués au sous-problème

$$(P) \equiv (B \mid x_j = 0 \ \forall j \in X_0 \ ; \ x_j = 1 \ \forall j \in X_1)$$

i.e.
$$\left[\begin{array}{l} \sum_{j\in X_1} c_j + \max \sum_{j\in X_2} c_j x_j \\ \quad \text{s.c.} \sum_{j\in X_2} a_j x_j \le b(P) = b - \sum_{j\in X_1} a_j \\ \quad x_j = 0 \text{ ou } 1 \quad \forall j \in X_2 \end{array}\right.$$

---

1 si $b(P) < 0$ alors $F(P) \equiv \emptyset$; aller en 12

2 si $b(P) = 0$ alors $x_j \leftarrow 0 \ \forall j \in X_2$
$X_0 \equiv X_0 \cup X_2$; $X_2 \equiv \emptyset$
aller en 5.1

3 $\forall j \in X_2 : a_j > b(P)$ alors $x_j \leftarrow 0$; $X_0 \equiv X_0 \cup \{j\}$
$X_2 \equiv X_2 \backslash \{j\}$

4 si $X_2 = \emptyset$ alors aller en 5.1

5 si $\sum_{j\in X_2} a_j \le b(P)$ alors $x_j \leftarrow 1 \ \forall j \in X_2$
$X_1 \equiv X_1 \cup X_2$; $X_2 \equiv \emptyset$
5.1 $v(P) \leftarrow \sum_{j\in X_1} c_j$
aller en 11

6 si $\bar{v}(P) \le \underline{v}(B)$ alors $\nexists x \in F(P) : cx > \underline{v}(B)$
aller en 12

7 Soit $x(P)$ la solution associée à $\bar{v}(P)$ (i.e. obtenue en résolvant une relaxation de (P))
si $x(P) \in F(P)$ alors $\underline{v}(B) \leftarrow cx(P)$
aller en 11.1

8 Utiliser une *heuristique* pour déterminer $\underline{v}(P)$.

9 si $\underline{v}(P) > \underline{v}(B)$ alors $\underline{v}(B) \leftarrow \underline{v}(P)$
si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$; stop.

10 Algorithme de réduction ⇒ modification de X_0, X_1, X_2
si $X_2 = \emptyset$ alors aller en 12
Descente dans l'arborescence

11 si $v(P) > \underline{v}(B)$ alors $\underline{v}(B) \leftarrow v(P)$
11.1 si $\underline{v}(B) = \bar{v}(B)$ alors $v(B) \leftarrow \underline{v}(B)$; stop
Algorithme de réduction

12 *Remontée dans l'arborescence*

Note

(i) La relaxation de (P) choisie est $(RP(u_i))$ c'est-à-dire $(RB(u_i) \mid x_j = 0 \ \forall j \in X_0 \ ; \ x_j = 1 \ \forall j \in X_1)$

(ii) Se référer à [K9] (et à [K11] pour une version récursive) pour l'explication des tests évoqués ci-dessus, dans le cadre de la mise en oeuvre de l'algorithme.

□

L'exemple numérique suivant donne le déroulement de notre méthode (appelée M.S.B.) lorsque m = 2 et la hiérarchie entre variables est donnée par le classement par ordre décroissant des valeurs absolues des coûts réduits à l'optimum de $(\bar{B})$.

Ce même exemple est repris pour le déroulement d'un algorithme de type Procédure par Séparation et Evaluation Séquentielle utilisant la technique énumérative dite L.I.F.O. (voir par exemple [G36]).

Les figure 6 et 7 donnent les parties de l'arborescence décrites par l'une et l'autre des méthodes (les numéros indiquent la chronologie des fixations de variables).

Considérons le problème :

$$(B)\quad \begin{cases} \max\ 52x_1 + 17x_2 + 40x_3 + 793x_4 + 813x_5 + 768x_6 + 724x_7 \\ \text{s.c. } 45x_1 + 15x_2 + 40x_3 + 793x_4 + 820x_5 + 777x_6 + 737x_7 \leq 837 \\ \quad x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,7 \end{cases}$$

Méthode M.S.B.

* *relaxation*

$\bar{x} = (1, 1, 1, \frac{737}{793}, 0, 0, 0)$, $v(\bar{B}) = 846$

$U = \{1, 2, 3\}$; $i = 4$; $L = \{5, 6, 7\}$

* *minorant de* $v(B)$

$x^2 = (1, 1, 1, 0, 0, 0, 1)$, $\underline{v}(B) = v_2(B) = 833$

* *majorant de* $v(B)$

$v(\bar{B} \mid x_4 = 0) = 839.708$; $v(\bar{B} \mid x_4 = 1) = 839 \Rightarrow \bar{v}(B) = 839$

* *réduction*

$v(RB(u_4) \mid x_7 = 1) = 833 = \underline{v}(B) \Longrightarrow x_7 \neq 0$

* *exploration lexicographique implicite :*

- *hiérarchie entre variables* : les variables sont considérées dans l'ordre décroissant des valeurs absolues des coûts réduits, c'est-à-dire :

$$d_6^1 = 9 > d_1^\circ = 7 \geq d_5^1 = 7 > d_2^\circ = 2 > d_3^\circ = 0 \geq d_4^\circ = 0$$

- *partition de* $\{1,\ldots,6\}$: $I = \{3,4\}$ et $\bar{I} = \{6, 1, 5, 2\}$

- *sommet initial* du cube unité de $\mathbb{R}^4$ pour l'exploration lexicographique :

(i) $x_{\bar{I}}^1 = [\bar{x}]_{\bar{I}} = (0, 1, 0, 1) = (\bar{x}_6, \bar{x}_1, \bar{x}_5, \bar{x}_2)$

$$\Rightarrow X_0 = \{6, 5\} \text{ et } X_1 = \{1, 2\} \Rightarrow (PA) \begin{cases} 69 + \max\ 40x_3 + 793x_4 \\ \text{s.c. } 40x_3 + 793x_4 \leq 777 \\ x_3, x_4 = 0 \text{ ou } 1 \end{cases}$$

$\Rightarrow x_4 = 0, x_3 = 1 \Rightarrow \underline{v(PA)} = 69 + 40 = \underline{109}\ (= v_1(B)) < \underline{v}(B)$

(ii) $x_{\bar{I}}^2 = (0, 1, 0, 0) \Rightarrow X_0 = \{6, 5, 2\}$ et $X_1 = \{1\}$

$$\Rightarrow (PA) \begin{cases} 52 + \max\ 40x_3 + 793x_4 \\ \text{s.c. } 40x_3 + 793x_4 \leq 792 \Rightarrow x_3 = 1, x_4 = 0 \Rightarrow \underline{v(PA)} = \\ x_3, x_4 = 0 \text{ ou } 1 \end{cases}$$

(iii) *modification de* x_5 ?

i.e. est-il utile de considérer les solutions telles que $x_6 = 0$ et $x_1 = x_5 = 1$:

$x_{\bar{I}}^3 = (0, 1, 1, 1)$ et $x_{\bar{I}}^4 = (0, 1, 1, 0)$?

c'est-à-dire

$$(P) \equiv (B \mid x_6 = 0\ ;\ x_1 = x_5 = 1) : \begin{cases} 865 + \max\ 17x_2 + 40x_3 + 793x_4 \\ \text{s.c. } 15x_2 + 40x_3 + 793x_4 \leq \\ x_2, x_3, x_4 = 0 \text{ ou } 1 \end{cases}$$

<u>non</u>, puisque $b(P) = -28 \Rightarrow F(P) = \emptyset$

(iv) *modification de* x_1 ?

i.e. est-il utile de considérer les solutions telles que $x_6 = 1$

$x_{\bar{I}}^5 = (0, 0, 0, 1)$

$x_{\bar{I}}^6 = (0, 0, 0, 0)$

$x_I^7 = (0, 0, 1, 1)$

$x_I^8 = (0, 0, 1, 0)$?

c'est-à-dire

$$(P) \equiv (B \mid x_6 = x_1 = 0) : \begin{cases} \max\ 813x_5 + 17x_2 + 40x_3 + 793x_4 \\ \text{s.c. } 820x_5 + 15x_2 + 40x_3 + 793x_4 \leq 837\ ? \\ x_j = 0 \text{ ou } 1\ ;\ j = 5, 2, 3, 4 \end{cases}$$

oui, puisque $\bar{v}(P) = v(RP(u_4)) = v(RB(u_4) \mid x_1 = 0) = 839 > \underline{v}(B)$.

$(v_2(P) = 830\ (x_5 = x_2 = 1,\ x_3 = x_4 = 0) \Rightarrow \underline{v}(P) < \underline{v}(B) = 833)$

$x_I^5 = (0, 0, 0, 1) \Rightarrow X_0 = \{6, 1, 5\}$ et $X_1 = \{2\}$

$$\Rightarrow (PA) \quad \begin{cases} 17 + \max\ 40x_3 + 793x_4 \\ \text{s.c. } 40x_3 + 793x_4 \leq 822 \Rightarrow x_3 = 0,\ x_4 = 1 \Rightarrow \underline{v(PA) = 810} \\ x_3, x_4 = 0 \text{ ou } 1 \end{cases}$$

(v) $x_I^6 = (0, 0, 0, 0) \Rightarrow X_0 = \{6, 1, 5, 2\}$ et $X_1 = \emptyset$

$$\Rightarrow (PA) \quad \begin{cases} \max\ 40x_3 + 793x_4 \\ \text{s.c. } 40x_3 + 793x_4 \leq 837 \Rightarrow x_3 = x_4 = 1 \Rightarrow \underline{v(PA) = 833} \\ x_3, x_4 = 0 \text{ ou } 1 \end{cases}$$

(vi) *modification de* x_5 ?

i.e. est-il utile de considérer x_I^7 et x_I^8, c'est-à-dire les solutions telles que $x_6 = x_1 = 0$ et $x_5 = 1$:

soit

$$(P) \equiv (B \mid x_6 = x_1 = 0\ ;\ x_5 = 1) : \begin{cases} 813 + \max\ 17x_2 + 40x_3 + 793x_4 \\ \text{s.c. } 15x_2 + 40x_3 + 793x_4 \leq 17\ ? \\ x_2, x_3, x_4 = 0 \text{ ou } 1 \end{cases}$$

non, puisque $\bar{v}(P) = v(P) = 813 + 17 = 830 < \underline{v}(B)$

vii) *modification de* x_6 ? (puisque toutes les solutions telles que $x_6 = 0$ ont été envisagées) :

i.e. est-il utile de considérer $x_{\bar{I}}^{9}$ à $x_{\bar{I}}^{16}$, c'est-à-dire les solutions telles que $x_6 = 1$:

soit

$$(P) \equiv (B \mid x_6 = 1) \quad \begin{cases} 768 + \max\ 52x_1 + 813x_5 + 17x_2 + 40x_3 + 793x_4 \\ \text{s.c. } 45x_1 + 820x_5 + 15x_2 + 40x_3 + 793\,x_4 \leq 60 \\ x_j = 0 \text{ ou } 1 \quad j = 1, 5, 2, 3, 4. \end{cases}$$

oui, puisque $v(RP(u_4)) = v(RB(u_4) \mid x_6 = 1) = 837 > \underline{v}(B)$

$x(P) = (1, 1, 0, 1, 0, 0) \in F(P) \Rightarrow \underline{v}(B) = v(P) = cx(P) = 837$

$\Rightarrow x_1^* = x_2^* = x_6^* = 1$ et $x_3^* = x_4^* = x_5^* = x_7^* = 0$.

□

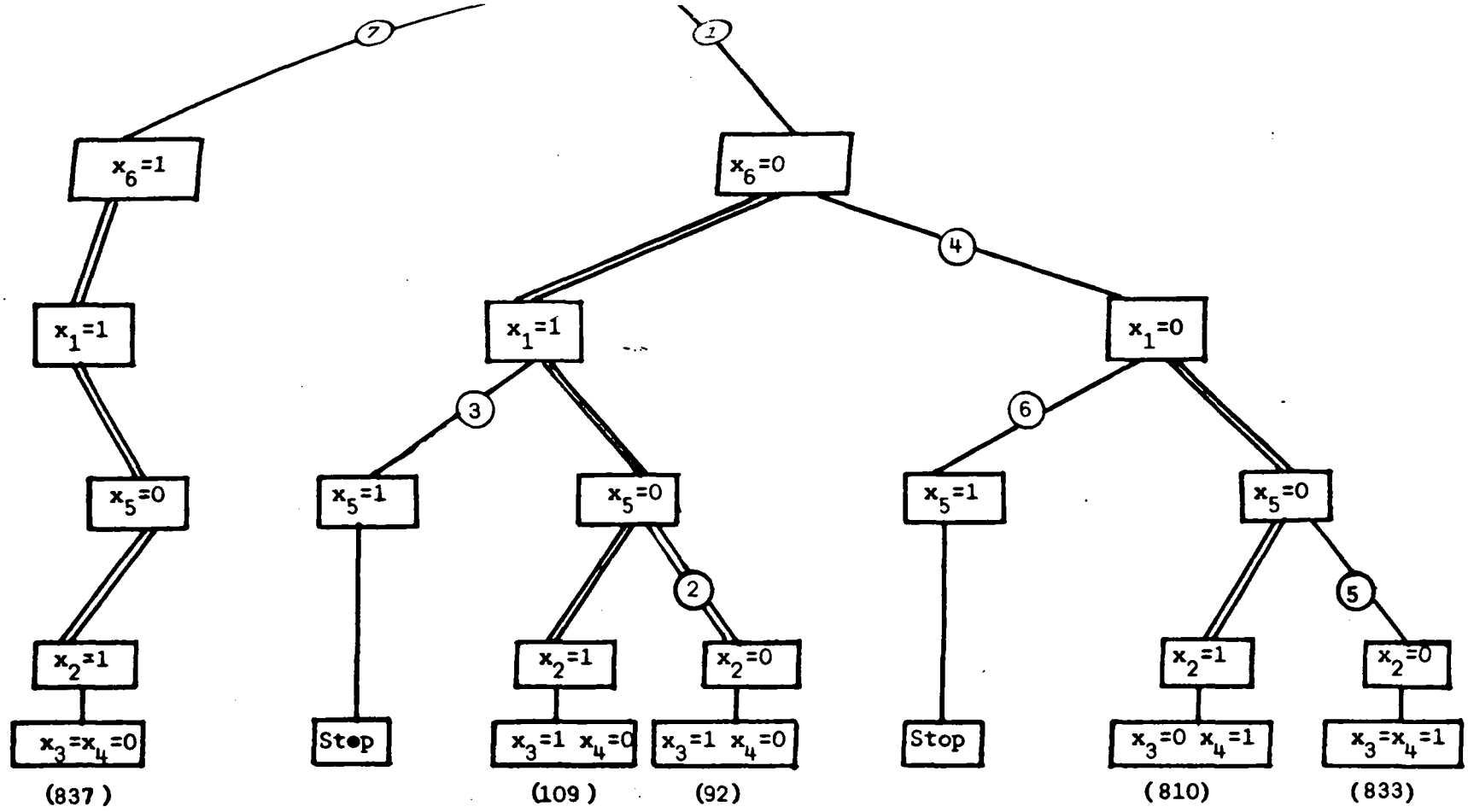


Figure 6

Les affectations des variables reliées par un trait $=$ sont faites globalement.

Méthode P.S.E.S

Cette méthode est liée à la technique énumérative dite LIFO qui consiste à gérer comme suit la liste d'indices PF des variables fixées à 0 ou 1 associée au déplacement dans l'arborescence :

- *descente dans l'arborescence*

(i) lorsqu'une variable x_j est fixée temporairement par un algorithme de réduction alors PF devient PF ∪ {k}.
[X_0 devient $X_0 \cup \{k\}$ si $x_k = 0$; X_1 devient $X_1 \cup \{k\}$ si $x_k = 1$]

(ii) le choix d'attribuer la valeur 1 à une variable x_k se traduit par le remplacement de PF par PF ∪ {k}. (X_1 devient $X_1 \cup \{k\}$)

- *remontée dans l'arborescence* : lorsqu'il est démontré l'inutilité de résoudre un sous-problème, la remontée jusqu'au noeud associé au choix se traduit par la recherche dans PF de l'indice non souligné situé le plus à droite pour

- d'une part, le souligner
- d'autre part, supprimer tous les indices situés à sa droite.

Les particularités de la méthode décrite (inspirée de celle de Greenberg et Hegerich [K23]) sont les suivantes :

1 tri des variables suivant l'ordre décroissant des rapport c_j/a_j

2 la *séparation* s'opère sur la variable d'indice j^*, telle que
$c_{j^*}/a_{j^*} = \max \{c_j/a_j \mid j \in X_2\}$
pour faire le *choix* d'attribuer la valeur 1 à x_{j^*}

3 *l'évaluation* par excès en un noeud utilise la valeur de la *relaxation* du sous-problème.

Ayant appliqué l'algorithme de réduction explicite pour l'algorithme M.S.B, on suppose donc avoir à explorer implicitement les solutions du problème

$$(B \mid x_7 = 0) \quad \begin{cases} \max\ 52x_1 + 17x_2 + 40x_3 + 793x_4 + 813x_5 + 768x_6 \\ \text{s.c. } 45x_1 + 15x_2 + 40x_3 + 793x_4 + 820x_5 + 777x_6 \leq 837 \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,6 \end{cases}$$

en sachant que $\bar{v}(B) = 839$ et $\underline{v}(B) = 833$

(i) *choix d'attribuer la valeur 1 à* x_1 : PF = {1} ($X_1 = \{1\}$, $X_0 = \emptyset$)

$$(P) \equiv (B \mid x_1 = 1) \quad \begin{cases} 52 + \max\ 17x_2 + 40x_3 + 793x_4 + 813x_5 + 768x_6 \\ \text{s.c. } 15x_2 + 40x_3 + 793x_4 + 820x_5 + 777x_6 \leq 792 \\ x_j = 0 \text{ ou } 1 \quad j = 2,\ldots,6 \end{cases}$$

$a_4 = 793 > b(P)$ et $a_5 = 820 > b(P) \Rightarrow$ $x_4 = x_5 = 0$ ($X_0 = \{4,5\}$)
PF = {1, 4, 5}

$\Rightarrow \bar{v}(P) = \lfloor v(\bar{B} \mid x_1 = 1, x_4 = x_5 = 0) \rfloor = 837 > \underline{v}(B)$

(ii) *choix d'attribuer la valeur 1 à* x_2 : PF = {1, 4, 5, 2} ($X_1 = \{1, 2\}$, $X_0 = \{4,5\}$)

$$\Rightarrow (P) \equiv (B \mid x_1 = x_2 = 1, x_4 = x_5 = 0) : \quad \begin{cases} 69 + \max\ 40x_3 + 768x_6 \\ \text{s.c. } 40x_3 + 777x_6 \leq 777 \\ x_3, x_6 = 0 \text{ ou } 1 \end{cases}$$

$\Rightarrow \bar{v}(P) = 837 > \underline{v}(B)$

(iii) *choix d'attribuer la valeur 1 à* x_3 : PF = {1, 4, 5, 2, 3} ($X_1 = \{1, 2, 3\}$, $X_0 = \{4, 5\}$)

$\Rightarrow$ $x_6 = 0$ et PF = {1, 4, 5, 2, 3, 6} ; la solution $\underline{x}$ associée est telle que $c\underline{x} = 109 < \underline{v}(B)$

(iv) *remontée dans l'arborescence* : PF = {1, $\underline{4}$, $\underline{5}$, 2, $\underline{3}$} ($X_1 = \{1, 2\}$, $X_0 = \{3, 4, 5\}$)

$\Rightarrow x_6 = 1$ et PF = {1, $\underline{4}$, $\underline{5}$, 2, $\underline{3}$, $\underline{6}$} ; la solution $\underline{x}$ associée est telle que $c\underline{x} = 837 > \underline{v}(B) \Rightarrow \underline{v}(B) = 837$

(v) *remontée dans l'arborescence* : PF = {1, $\underline{4}$, $\underline{5}$, $\underline{2}$} ($X_1 = \{1\}$, $X_0 = \{2, 4, 5\}$)

$$\Rightarrow (P) \equiv (B \mid x_1 = 1, x_2 = x_4 = x_5 = 0) : \begin{cases} 52 + \max\ 40x_3 + 768x_6 \\ \text{s.c. } 40x_3 + 777x_6 \leq 792 \\ x_3, x_6 = 0 \text{ ou } 1 \end{cases}$$

$\Rightarrow \bar{v}(P) = \lfloor v(\bar{P}) \rfloor = 835 < \underline{v}(B)$; il est donc inutile de résoudre (P)

(vi) *remontée dans l'arborescence* : PF = {$\underline{1}$} ($X_1 = \emptyset$; $X_0 = \{1\}$)

$$\Rightarrow (P) \equiv (B \mid x_1 = 0) \quad \begin{cases} \max\ 17x_2 + 40x_3 + 793x_4 + 813x_5 + 768x_6 \\ \text{s.c. } 15x_2 + 40x_3 + 793x_4 + 820x_5 + 777x_6 \leq 837 \\ x_j = 0 \text{ ou } 1 \quad j = 2, \ldots, 6 \end{cases}$$

$\Rightarrow \bar{v}(P) = \lfloor v(\bar{B} \mid x_1 = 0) \rfloor = 839 > \underline{v}(B)$

$v(RP\ (1) \mid x_2 = 0) = 837 = \underline{v}(B) \Rightarrow x_2 \leftarrow 1$
$v(RP\ (1) \mid x_5 = 1) = 832 < \underline{v}(B) \Rightarrow x_5 \leftarrow 0$
$v(RP\ (1) \mid x_6 = 1) = 830 < \underline{v}(B) \Rightarrow x_6 \leftarrow 0$

sous l'hypothèse $x_1 = 0$

$\Rightarrow$ PF = {$\underline{1}$, $\underline{2}$, $\underline{5}$, $\underline{6}$} ($X_1 = \{2\}$, $X_0 = \{1, 5, 6\}$)

(vii) *choix d'attribuer la valeur 1 à* x_3 : PF = {$\underline{1}$, $\underline{2}$, $\underline{5}$, $\underline{6}$, 3} ($X_1 = \{2, 3\}$, $X_0 = \{1, 5, 6\}$)

$\Rightarrow x_4 = 0 \Rightarrow$ PF = {$\underline{1}$, $\underline{2}$, $\underline{5}$, $\underline{6}$, 3, $\underline{4}$} ; la solution $\underline{x}$ associée est telle que $c\underline{x} = 57 < \underline{v}(B)$

(viii) *remontée dans l'arborescence* : PF = {1, 2, 5, 6, 3} ($X_1 = \{2\}$, $X_0 = \{1, 3, 5, 6\}$)

$\Rightarrow x_4 = 1 \Rightarrow$ PF = {1, 2, 5, 6, 3, 4} ; la solution $\underline{x}$ associée est telle que $c\underline{x} = 810 < \underline{v}(B)$.

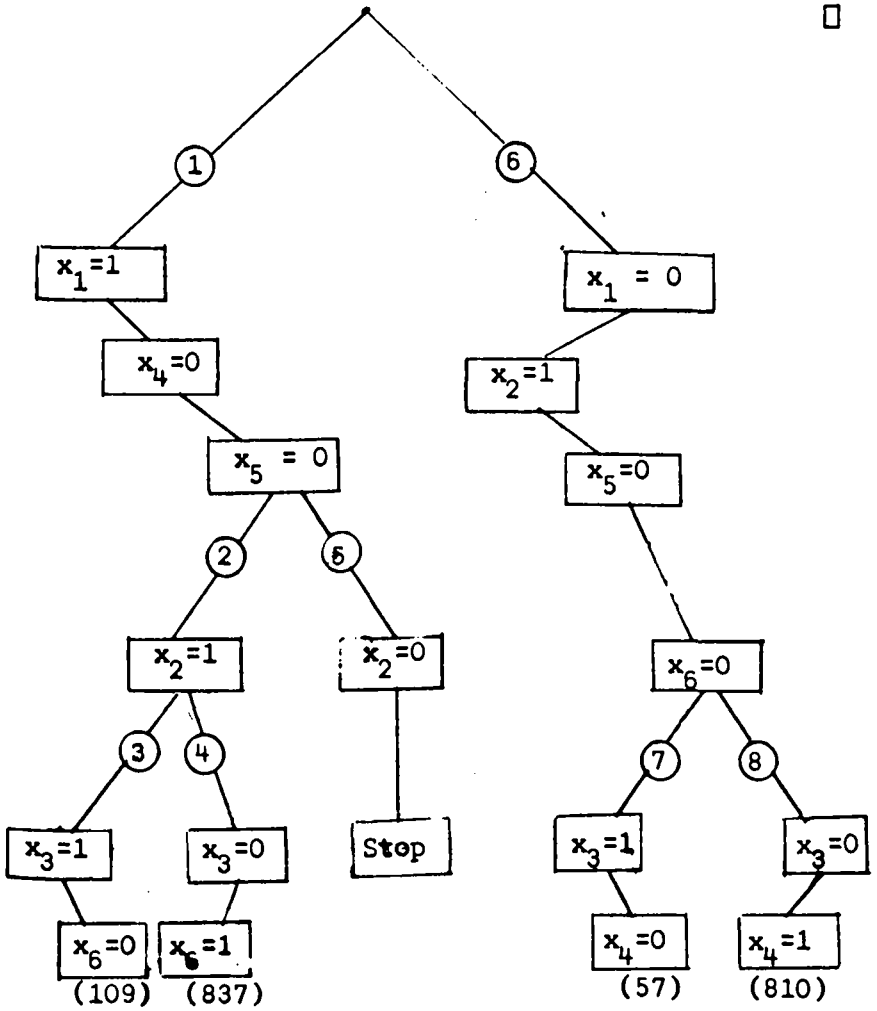


Figure 7

VII - EXPERIENCES NUMERIQUES

Les algorithmes présentés en parties II, V et VI ont été mis en oeuvre sur UNIVAC 1110 en considérant 534 problèmes tests :

- des problèmes de 50, 100, 500, 1000, 2000, 5000, 7500, 9000 et 13000 variables tirés au hasard suivant la loi uniforme

$$(c_j, a_j \in [0,99] \text{ et } b \in [\max_{1\le j\le n} a_j, \sum_{j=1}^{n} a_j[),$$

à raison de 50 exemplaires pour chaque taille.

- les 84 problèmes tests de 75, 100, 150 et 200 variables provenant de l'Ecole des Mines du Colorado [K13].

Pour chaque série de problèmes n (resp. NP) désignera la taille (resp. le nombre) des problèmes de la série. Tous les temps sont donnés en millisecondes.

A titre d'information, les tableaux 11 et 12 fournissent un résumé des principales expériences de la littérature (les années de parution de chaque article sélectionné sont répertoriées dans le tableau 10) :

- le tableau 11 indique pour chaque méthode les quantités :
  $\bar{n}$: taille maximale envisagée
  $\overline{NP}$: nombre d'exemplaires dans la taille $\bar{n}$

- le tableau 12 donne les caractéristiques des problèmes traités, tous tirés au hasard suivant la loi uniforme.

Auteurs		Référence	Année	Auteurs		Référence	Année
BR	Barr et Ross	K3	75	IK	Ingorgiola et Korsh	K28	73
C	Cabot	K4	70	L	Laurière	K31	78
DH	Dembo et Hammer	K6	74	MT	Martello et Toth	K35	78
FP1	Fayard et Plateau	K8,K10	71	N	Nauss	K37	76
FP2	Fayard et Plateau		79	S	Suhl	K41	78
H	Hansen	K26	74	Z	Zoltners	K45	78
HS	Horowitz et Sahni	K27	74				

Tableau 10

$\overline{NP}$ \ $\overline{n}$	20	60	200	500	1000	5000	7500	9000	60000
5		HS			MT				
9			N						
10				H					
25						DH Z	BR		
27								S	
40					IK				
50	C						FP2		L
100					FP1				

Tableau 11

c_j , a_j ϵ		]0, 100] ou [10, 100]	]0, 1000]	$c_j \epsilon [a_j-\beta, a_j+\beta]$
$\max_j a_j \leq b \leq \Sigma a_j$		FP1 , FP2 S		
$b = \alpha\Sigma a_j$	$\alpha\epsilon]0,1[$	H L MT		
	$\alpha = 0.2$	HS, FP2	HS	HS , MT (β=0) (β=10)
	$\alpha = 0.5$	BR,DH FP2,HS IK,N Z	BR HS Z	HS , MT (β=0) (β=10)
	$\alpha = 0.8$	FP2		MT (β=10)

Tableau 12

VII.1 - RELAXATION

VII.1.1 Mise en oeuvre

La mise en oeuvre des algorithmes CKR (utilisation du "tri rapide" (Quick sort)) et NKR nécessite la sélection d'un *élément de référence* $k \in J$ donné ; parmi différentes solutions possibles (voir [K13]), notre choix s'est porté sur l'élément "médian" des trois éléments situés en tête, en fin et au milieu du fichier à partitionner.

Note

Dans le cadre des problèmes tirés au hasard, des résultats relatifs à l'introduction d'une évaluation préalable du rapport c_i/a_i sont répertoriés au paragraphe VII.1.2e. sous le vocable NKR EVAL.

□

D'autre part à partir d'une certaine taille de fichier, appelée *Seuil* par la suite, l'algorithme de tri rapide, moins performant, est à remplacer par le tri dit par insertion [C21]. Ce seuil (10 pour NKR, 8 pour CKR) a été établi expérimentalement sur 10 problèmes de 5000 variables (tableau 13).

Enfin, J étant donné, lors de la détermination de la (R,k) - partition, les éléments de E(J,k) sont répartis dans U(J,k) et L(J,k) de manière à minimiser les permutations. La mise en oeuvre de l'algorithme CKR est proche de celle préconisée pour le tri rapide par Sedgewick [C21]

Seuil	CKR	NKR
5	974.71	272.96
6	963.15	-
7	965.59	-
8	959.41	272.98
9	965.43	272.67
10	969.47	270.70
11	970.58	273.26
15	1009.36	300.64

Tableau 13

VII.1.2 Résultats numériques

a) Le tableau 14 donne, pour les problèmes tirés au hasard, un aperçu des temps de calcul relatifs aux algorithmes CKR select (CKR avec une méthode de tri dite par sélection [C1, C15], utilisée dans la méthode MSB 71 [K8, K10]), CKR Quicks (CKR avec la méthode de tri rapide) et NKR (voir partie II).

La colonne gain % CKR Quicks (resp. gain % CKR select) indique le pourcentage moyen de gain de l'algorithme NKR par rapport aux méthodes classiques.

b) Les tableaux 15 et 16 donnent une idée de la dispersion des temps de calcul ; ils fournissent, pour chaque taille de problème, en colonne

- Min (resp. Max) : le temps de résolution minimum (resp. maximum)

- Moy. Min(resp. Moy. Max) : la moyenne des dix plus faibles (resp. importants) temps de résolution

- Moyenne : le temps moyen de 50 problèmes tests.

n	NP	NKR	CKR Quicks	CKR Sélect	Gain % CKR Quicks	Gain % CKR Select
50	50	3.02	5.23	14.96	42.26 %	79.81 %
100	50	5.78	11.50	51.85	49.74 %	88.85 %
500	50	26.47	73.07	1 135.79	63.77 %	97.67 %
1000	50	48.10	159.42	4 451.53	69.83 %	98.92 %
2000	50	102.10	345.87	16 500 *	70.48 %	99.38 %
5000	50	270.70	959.41	/	71.78 %	/
9000	50	478.68	1849.80	/	74.12 %	/

* uniquement pour 3 problèmes tests.

Tableau 14

CKR	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
50	4.6	4.86	5.23	5.60	5.8
100	10.6	10.86	11.50	12.08	13.0
500	66.0	69.48	73.07	77.26	81.0
1000	131.6	150.78	159.42	167.44	174.6
2000	321.8	333.78	345.87	358.76	376.8
5000	907.4	927.44	959.41	995.52	1 039.2
9000	1 769.0	1 797.54	1 849.80	1 898.88	1 932.8

Tableau 15

NKR	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
50	1.8	2.06	3.02	4.08	5.2
100	2.6	4.06	5.78	7.56	8.8
500	11.0	17.50	26.47	36.52	39.0
1000	27.2	32.02	48.10	66.38	80.4
2000	66.8	74.90	102.10	136.42	170.8
5000	139.4	189.04	270.70	359.08	405.2
9000	274.0	323.02	478.68	612.94	696.4

Tableau 16

c) Les temps indiqués pour les algorithmes NKR et CKR Quicks dans les tableaux 13 à 16 concernent une version "rapide" d'encombrement mémoire 6n pour CKR Quicks et 5n pour NKR. Les tableaux 17, 18 et 19 sont eux relatifs à des versions moins rapides (CKR.L, NKR.L), mais d'encombrement mémoire 4n.

Note

Il serait possible d'atteindre un encombrement mémoire 3n en supprimant la sauvegarde de la numérotation d'origine des données.

□

n	NP	NKR.L	CKR.L	gain % CKR.L
1000	50	49.98	224.16	77.70 %
5000	50	285.54	1 360.46	79.01 %
9000	50	508.82	2 504.82	79.69 %
13000	50	758.79	3 900.07	80.54 %

Tableau 17

CKR.L	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
1000	178.8	209.92	224.16	235.28	243.8
5000	1 308.6	1 324.80	1 360.46	1 410.06	1 488.4
9000	2 100.01	2 348.41	2 504.82	2 584.63	2 709.2
13000	3 427.6	3 767.28	3 900.07	4 033.20	4 185.2

Tableau 18

NKR.L	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
1000	24.2	30.98	49.98	71.56	87
5000	111	188.80	285.54	387.82	434.2
9000	262.6	334.06	508.82	661.36	774.4
13000	116.2	444.26	758.79	1 121.92	1 361.8

Tableau 19

d) Le tableau 20 contient pour chacune des sept séries de problè
de l'Ecole des Mines du Colorado

- les moyennes des temps de calcul de

  . l'algorithme CKR en utilisant la méthode de tri Select (colonne Alg. CKR/Select) et la méthode de tri Quicksort (colonne Alg. CKR/Quicks)

  . l'algorithme NKR (colonne Alg. NKR/Moy.)

- le temps minimum (resp. maximum) atteint pour résoudre la re
laxation avec l'algorithme NKR (colonne Alg. NKR/Min (resp. Alg. NKR/Max

- les comparaisons des algorithmes NKR et CKR :

  . colonne Gain % CKR/Select : rapport moyen des temps de l'algorithme CKR en utilisant Select sur les temps de l'al
gorithme NKR

  . colonne Gain % CKR/Quicks : pourcentage moyen des gains obtenus par l'utilisation de l'algorithme NKR au lieu de l'
gorithme CKR (avec Quicksort pour le tri).

Séries	n	NP	Alg. CKR		Alg. NKR			Gain % CKR	
			Select	Quicks	Min	Moy	Max	Select	Quicks
1									
2	200	10	170.30	36.05	10.2	14.68	22.6	11.60	59.26
3	100	10	45.56	14.76	2.4	5.32	8.1	8.56	63.96
4	100	10	45.65	14.55	4.1	8.36	11.2	5.46	42.54
5	100	10	45.60	13.70	5.1	6.14	7.1	7.43	55.18
6	100	10	45.70	14.50	4.3	7	8.3	6.53	51.72
7	150	12	96	25.94	3.1	13.05	17.4	7.36	49.69
	75	22	26.30	11.40	3.2	4.63	5.4	5.68	59.39

Tableau 20

e) *Algorithme NKR EVAL :*

Pour les problèmes dont les données sont tirées au hasard il est relativement aisé d'obtenir une bonne évaluation de la valeur du rapport de base c_i/a_i. Cette évaluation préalable permet d'améliorer très sensiblement (jusqu'à 52 % pour n = 5000) les performances de l'algorithme NKR.

(i) *détermination de l'évaluation* c_{i_e}/a_{i_e} *de* c_i/a_i :

Cette évaluation est basée sur la détermination d'un encadrement $[\underline{i}, \bar{i}]$ de l'indice de base i, suivie d'une intégration de la fonction densité de probabilité de la variable aléatoire R = C/A où C (resp. A) est la variable aléatoire relative à la fonction économique (resp. contrainte).

* *Encadrement de la valeur de l'indice de base*

- estimation par excès $\bar{I}$ de i :

Les données du problème étant supposées classées dans l'ordre des valeurs décroissantes des c_j/a_j, la détermination de l'estimation par excès $\bar{I}$ de i est basée sur la construction d'une contrainte $a'x \leq b$ telle que

$$\sum_{j=1}^{i} a'_j \leq \sum_{j=1}^{i} a_j .$$

L'intervalle [0,100] est subdivisé en m intervalles contenant chacun $p(\simeq \frac{n}{m})$ éléments pour définir a'x ≤ b par

$$\sum_{k=1}^{m} \sum_{j \in J_k} a'_j x_j \leq b$$

$$\text{avec } \forall k \in \{1,\ldots,m\} \begin{cases} J_k = \{(k-1)\,p+1,\ (k-1)\,p+2,\ldots,kp\} \\ a'_j = \frac{100}{2m}\,(2k-1) \quad \forall j \in J_k \end{cases}$$

Les déterminations de

$$\cdot\ k^* : \sum_{\ell=1}^{k^*-1} \sum_{j \in J_\ell} a'_j \leq b < \sum_{\ell=1}^{k^*} \sum_{j \in J_\ell} a'_j$$

$$\cdot\ i_k \in J_{k^*} : \sum_{\ell=1}^{k^*-1} \sum_{j \in J_\ell} a'_j + \sum_{j=(k^*-1)p+1}^{i_k-1} a''_j \leq b$$

$$\text{et } b < \sum_{\ell=1}^{k^*-1} \sum_{j \in J_\ell} a'_j + \sum_{j=(k^*-1)p+1}^{i_k} a''_j$$

$$\text{avec } a''_j = \begin{cases} \frac{100}{2m}\,(2k^* - \frac{3}{2}) \quad \forall j \in J_1 = (k^*-1)p+1,\ldots,(k^*-1)p+\left\lfloor \frac{p}{2} \right\rfloor \\ \frac{100}{2m}\,(2k^* - \frac{1}{2}) \quad \forall j \in J_{k^*} \setminus J_1 \end{cases}$$

permettent de poser :

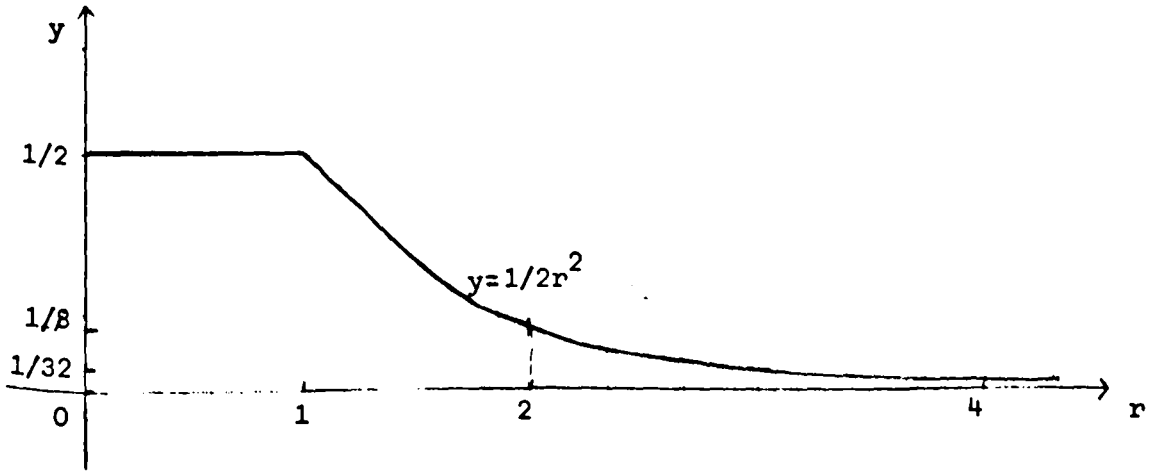
$$\bar{i} = (k^*-1)\,p + i_k$$

- estimation par défaut $\underline{i}$ de i :

Elle est obtenue en supposant que $\forall j\ a_j = 50 \Rightarrow \underline{i} = b/50$

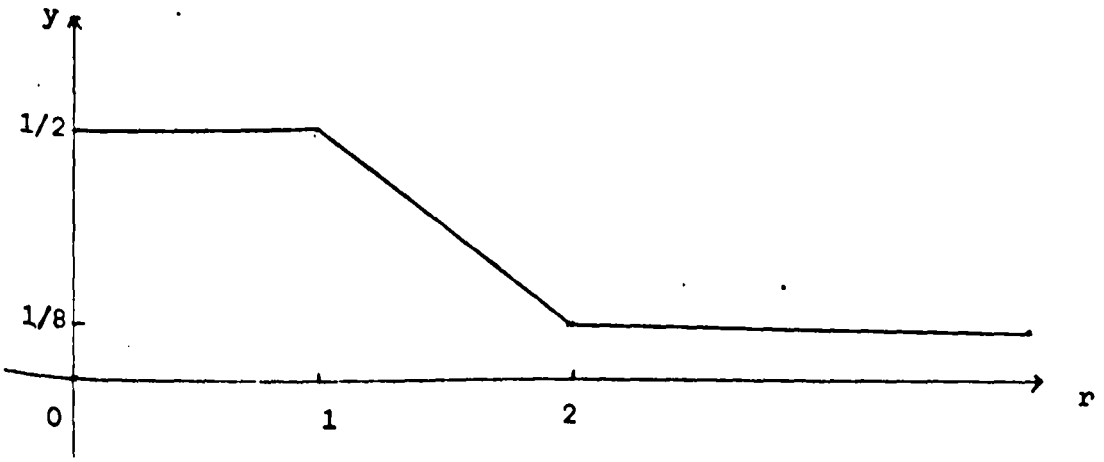
* Evaluation de c_i/a_i

Soit γ la courbe représentative de la fonction densité de probabilité de R



Connaissant $\bar{I}$ et $\underline{i}$ il est aisé de calculer par intégration $c_{\bar{I}}/a_{\bar{I}}$. évaluation par défaut de c_i/a_i et $c_{\underline{i}}/a_{\underline{i}}$ évaluation par excès de c_i/a_i.

Pratiquement nous avons choisi d'approcher γ par la courbe suivante pour poser $c_{i_e}/a_{i_e} = (2\, c_{\bar{I}}/a_{\bar{I}} + c_{\underline{i}}/a_{\underline{i}})/3$



(ii) *Résultats numériques (m=5)*

La colonne "% erreur" du tableau 21 qui donne l'erreur moyenne faite lors de l'évaluation de c_i/a_i pour les différentes tailles de problème, montre la précision de cette méthode ; le tableau 22 traduit son efficacité

par comparaison avec les temps d'exécution de l'algorithme NKR.L (NKR EVAL est construit à partir de NKR.L).

f) Finalement, une comparaison des cinq algorithmes décrits ci-dessus est faite dans les tableaux 23, 24 et 25 respectivement pour les problèmes de 1000, 5000 et 9000 variables.

n	$\underline{i}$	$\bar{i}$	i	c_{ie}/a_{ie}	c_i / a_i	% erreur
50	23.58	32.24	30.46	1.0042	0.8504	18.09 %
100	48.46	66.14	60.80	0.9408	0.8345	12.74 %
500	226.38	315.06	287.14	1.0471	1.0486	0.14 %
1000	501.36	661	615	0.9930	1.1591	14.33 %
5000	2 540.16	3 117.96	3 347.88	0.9476	0.8428	12.43 %
9000	3 046	4 032	3 729	0.5919	0.5382	9.98 %

Tableau 21

n	NKR EVAL	NKR.L	% gain
1000	36.88	49.98	35.52 %
5000	187.32	285.54	52.43 %
9000	344.18	508.82	47.84 %

Tableau 22

Méthode	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
NKR EVAL	20.2	24.84	36.88	49.30	58.6
NKR	27.2	32.02	48.10	66.38	80.4
NKR.L	24.2	30.98	49.98	71.56	87
CKR	131.6	150.78	159.42	167.44	174.6
CKR.L	178.8	209.92	224.16	235.28	243.8

Tableau 23 - 1000 variables

Méthode	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
NKR EVAL	97.8	140.9	187.32	233.28	279.4
NKR	139.4	189.04	270.70	359.08	405.2
NKR.L	111	188.80	285.54	387.82	434.2
CKR	907.4	927.44	959.41	995.52	1039.2
CKR.L	1308.6	1324.80	1360.46	1410.06	1488.4

Tableau 24 - 5000 variables

Méthode	MIN	MOY. MIN	Moyenne	MOY. MAX	MAX
NKR EVAL	198.0	262.16	344.18	433.24	493
NKR	274	323.02	478.68	612.94	696.4
NKR.L	262.6	334.06	508.82	661.36	774.4
CKR	1769	1797.54	1849.8	1898.88	1932.8
CKR.L	2150.21	2399.22	2504.82	2599.77	2672.26

Tableau 25 - 9000 variables

VII.2 - RÉDUCTION

L'algorithme de réduction de la partie V a été mis en oeuvre

- pour des problèmes tirés au hasard : résultats repertoriés dans les tableaux 27 (problèmes 1 à 13) et 31.

- pour les problèmes de la littérature suivants :

(i) 19 problèmes caractérisés dans le tableau 26

(ii) 84 problèmes de l'école des Mines du Colorado résultats répertoriés dans les tableaux 27 (problèmes 14 à 32), 28, 29, 30, et 33 ;

Problème	Domaine des c_j	Domaine des a_j	b	Référence
14			999	
15			2000	
16	[1,789]	[1,3541]	3000	Chapitre 4 de [K9]
17			5000	
18			7000	
19			8000	
20			12000	
21		[1,280]	600	Contraintes 1, 2 et 3 du problème 6 de [G48]
22	[22,4260]	[1,200]	500	
23		[1,100]	500	
24	[20,101]	[2,10]	11	
25	[5,15]	[4,8]	17	
26	[6,11]	[2,10]	20	Problèmes 1 à 9 du § III.4
27	[6,11]	[2,10]	20	
28	[17,813]	[15,820]	837	
29	[11,65]	[1,55]	110	
30	[2,457]	[5,631]	650	
31	[1,20]	[1,30]	90	
32	[5,15]	5,13	51	

Tableau 26

a) Les tableaux 27, 28 et 29 rassemblent en colonne :

n : la taille du problème

$\underline{v}_1$ (resp. $(\bar{v}_1)$) : l'évaluation par défaut (resp. excès)obtenue à l'étape 3 (resp. 4) de la méthode de réduction

ITER j (j=1,...,4) : le nombre de variables éliminées au cours de la jème itération de l'étape 5

L : le nombre de variables éliminées après l'itération 1, en appliquant le principe de l'algorithme de Laurière [K31] ; le nombre total de variables éliminées par cet algorithme est donné par la somme des nombres des colonnes ITER 1 et L

$\underline{v}_2$ (resp. $(\bar{v}_2)$) : l'évaluation par défaut (resp. excès) trouvée en fin d'itération 2 de l'étape 5

NVR : le nombre de variables "restantes" (i.e. non éliminées) à la fin de notre méthode de réduction

Note

(i) Les valeurs indiquées dans les colonnes $\underline{v}_j$ et $\bar{v}_j$(j=1,2) sont soulignées lorsqu'elles atteignent la valeur optimale du problème.

(ii) Pour les tableaux 28 et 29 on trouvera en colonne

DOM : le nombre de variables éliminées par les relations de dominance (au cours de la première itération de l'étape 5)

(iii) - Les valeurs optimales des problèmes 19 et 24 (tableau 27) (respectivement 5899 et 108) ne sont atteintes qu'au cours de l'itération 4

- Il en est de même pour les problèmes 5.8, 6.3 et 7.21 (tableau 29) (respectivement 4150, 5793 et 12528)

(iv) Pour les problèmes de l'Ecole des Mines du Colorado, seules les valeurs de dix problèmes ne sont pas atteintes en fin de méthode de réduction le tableau 30 répertorie ces valeurs et les écarts avec les évaluations par défaut de fin d'algorithme.

□

Pb	n	$\underline{v}_1$	$\bar{v}_1$	ITER 1	L	ITER 2	$\underline{v}_2$	$\bar{v}_2$	ITER 3	ITER 4	NVR
1	50	1286	1319	24	18	18	1313	1318	2	6	0
2	50	1060	1088	35	2	2	1062	1085	5	0	8
3	50	1814	1844	24	7	7	1817	1844	6	0	13
4	50	2007	2012	44	1	6	2007	2012	-	-	0
5	50	1763	1786	28	14	14	1778	1785	8	-	0
6	50	1449	1457	43	2	7	1451	1456	-	-	0
7	50	1554	1577	38	8	12	1565	1576	-	-	0
8	100	3279	3294	68	20	20	3289	3294	2	0	10
9	100	2946	2953	88	0	0	2946	2951	5	1	6
10	100	4612	4631	65	27	35	4628	4631	-	-	0
11	500	22890	22891	496	4	4	22890	22890	-	-	0
12	500	7012	7016	488	10	12	7014	7016	-	-	0
13	500	7282	7291	467	19	19	7286	7291	9	5	0
14	40	4136	4249	19	2	9	4190	4214	6	6	0
15	40	4653	4789	17	5	23	4749	4751	-	-	0
16	40	4983	5002	4	31	36	4983	4999	-	-	0
17	40	5162	5513	5	1	35	5321	5344	-	-	0
18	40	5826	5834	35	5	5	5826	5834	-	-	0
19	40	5894	5916	31	0	0	5897	5912	9	-	0
20	40	6081	6085	36	2	4	6081	6083	-	-	0
21	37	10780	10965	7	8	23	10883	10920	-	-	0
22	34	10129	10706	1	8	33	10537	10566	-	-	0
23	32	12343	12347	30	2	2	12343	12347	-	-	0
24	4	101	111	0	0	0	101	110	0	4	0
25	4	28	32	1	3	3	30	31	-	-	0
26	4	24	32	0	3	4	29	32	-	-	0
27	5	24	32	0	2	5	29	32	-	-	0
28	7	833	846	1	2	6	837	839	-	-	0
29	8	139	164	0	4	8	159	164	-	-	0
30	9	313	508	0	9	9	471	471	-	-	0
31	10	67	77	0	5	10	75	76	-	-	0
32	12	73	78	5	1	1	75	77	1	5	0

Tableau 27

Tableau 28

Pb	n	$\underline{V}_1$	$\bar{V}_1$	ITER 1	DOM	L	ITER 2	$\underline{V}_2$	$\bar{V}_2$	ITER 3	ITER 4	NVR
1.1	200	2123	2151	169	152	29	31	2147	2151	-	-	0
1.2	200	3763	3771	181	165	8	8	3766	3771	1	1	9
1.3	200	5579	5589	166	150	17	17	5584	5589	2	1	14
1.4	200	6585	6586	196	185	4	4	6585	6586	-	-	0
1.5	200	7487	7490	190	178	10	10	7488	7489	-	-	0
1.6	200	8308	8314	178	162	19	22	8311	8313	-	-	0
1.7	200	9016	9024	164	154	34	46	9023	9024	-	-	0
1.8	200	9661	9662	198	191	2	2	9661	9662	-	-	0
1.9	200	10601	10606	181	170	10	10	10604	10605	1	1	7
1.10	200	11118	11118	200	194	200	-	-	-	-	-	0
2.1	100	1135	1165	86	76	6	7	1148	1163	7	-	0
2.2	100	1701	1709	88	78	4	7	1701	1706	2	3	0
2.3	100	2401	2415	78	65	19	22	2411	2415	-	-	0
2.4	100	2985	2988	95	87	2	5	2985	2987	-	-	0
2.5	100	3451	3463	75	63	18	25	3459	3463	-	-	0
2.6	100	3859	3865	86	73	1	1	3859	3865	3	0	10
2.7	100	4203	4210	86	75	7	7	4207	4210	1	6	0
2.8	100	4489	4491	97	92	3	3	4489	4491	-	-	0
2.9	100	4674	4676	97	88	3	3	4674	4676	-	-	0
2.10	100	4805	4805	-	-	-	-	-	-	-	-	0
3.1	100	1319	1326	95	88	4	5	1319	1325	-	-	0
3.2	100	2016	2021	91	84	3	3	2016	2020	2	4	0
3.3	100	2593	2597	93	85	1	1	2593	2596	3	3	0
3.4	100	3087	3107	64	52	32	36	3102	3105	-	-	0
3.5	100	3549	3577	52	38	44	48	3573	3577	-	-	0
3.6	100	4004	4023	60	48	20	20	4011	4023	5	2	13
3.7	100	4414	4443	49	31	50	51	4438	4442	-	-	0
3.8	100	4783	4805	60	44	31	31	4801	4805	2	1	6
3.9	100	5430	5435	91	81	4	9	5430	5435	-	-	0
3.10	100	5908	5922	78	70	14	25	5916	5921	7	-	0
4.1	100	719	723	91	82	7	9	722	723	-	-	0
4.2	100	1358	1373	65	58	34	35	1371	1373	-	-	0
4.3	100	1952	1961	70	60	23	23	1957	1960	3	4	0
4.4	100	2491	2498	78	63	12	12	2494	2497	2	2	6
4.5	100	2992	3000	72	58	11	11	2995	3000	5	4	8
4.6	100	3468	3473	86	73	1	1	3468	3472	2	1	10
4.7	100	3911	3927	40	29	35	35	3921	3927	3	4	18
4.8	100	4358	4361	86	72	5	5	4359	4361	3	0	6
4.9	100	5152	5166	53	43	41	41	5164	5166	2	4	0
4.10	100	5850	5861	62	51	38	38	5860	5861	-	-	0

Pb	n	$\underline{V}_1$	$\bar{V}_1$	ITER 1	DOM	L	ITER 1	$\underline{V}_2$	$\bar{V}_2$	ITER 3	ITER 4	NVR
5.1	100	430	492	65	55	32	35	484	491	-	-	0
5.2	100	970	997	68	59	23	23	990	996	1	0	8
5.3	100	1705	1710	91	80	80	9	1708	1710	-	-	0
5.4	100	2340	2344	93	82	0	0	2340	2344	2	5	0
5.5	100	2898	2905	89	77	4	4	2900	2905	2	0	5
5.6	100	3373	3384	80	67	18	18	3380	3384	2	-	0
5.7	100	3794	3799	91	82	4	4	3794	3799	3	3	0
5.8	100	4142	4151	82	76	10	10	4148	4151	0	8	0
5.9	100	4438	4443	89	80	7	11	4441	4443	-	-	0
5.10	100	4659	4663	90	83	2	4	4659	4663	1	5	0
6.1	150	4264	4264	-	-	-	-	-	-	-	-	0
6.2	150	5059	5076	119	105	30	31	5074	5076	-	-	0
6.3	150	5763	5796	92	76	51	53	5792	5795	2	3	0
6.4	150	6457	6481	102	91	40	40	6475	6477	2	6	0
6.5	150	7094	7128	82	66	59	59	7122	7128	0	1	8
6.6	150	7701	7706	140	127	10	10	7706	7706	-	-	0
6.7	150	8192	8217	84	70	21	44	8199	8208	3	1	18
6.8	150	8641	8717	28	21	24	112	8680	8684	10	-	0
6.9	150	9666	9681	110	96	31	31	9675	9680	5	4	0
6.10	150	10475	10483	129	118	21	21	10480	10482	-	-	0
6.11	150	11180	11188	128	119	13	13	11185	11188	0	0	9
6.12	150	11709	11715	132	122	18	18	11714	11714	-	-	0
7.1	75	5696	5716	56	45	0	10	5696	5709	2	2	5
7.2	75	6334	6352	59	48	2	16	6334	6338	-	-	0
7.3	75	6794	6954	23	19	15	46	6933	6942	6	-	0
7.4	75	7487	7510	54	42	1	9	7487	7502	0	0	12
7.5	75	7579	7610	44	32	10	27	7585	7595	1	3	0
7.6	75	7668	7710	40	29	4	24	7670	7687	2	9	0
7.7	75	7770	7810	40	29	2	29	7770	7780	6	-	0
7.8	75	7853	7910	35	26	4	29	7859	7872	3	8	0
7.9	75	7918	8010	27	22	13	48	7964	7905	-	-	0
7.10	75	8408	8510	25	20	40	45	8496	8507	5	-	0
7.11	75	8970	8980	64	51	4	11	8970	8974	-	-	0
7.12	75	9053	9072	56	44	5	8	9059	9062	3	8	0
7.13	75	9118	9165	38	26	37	37	9164	9164	-	-	0
7.14	75	9242	9257	59	48	3	4	9242	9257	0	0	12
7.15	75	9334	9348	60	49	9	12	9340	9346	0	3	0
7.16	75	9423	9439	59	48	5	5	9425	9428	2	9	0
7.17	75	10164	10248	28	21	47	47	10245	10247	-	-	0
7.18	75	10514	10611	27	20	30	40	10592	10607	3	0	5
7.19	75	11558	11572	66	60	5	9	11561	11566	-	-	0
7.20	75	12021	12174	21	14	20	54	12065	12068	-	-	0
7.21	75	12204	12620	8	6	8	20	12424	12557	1	46	0
7.22	75	12864	12896	56	48	17	19	12878	12890	-	-	0

Tableau 29

Pb	3.6	4.4	4.5	4.6	4.7	6.7	7.4	7.6	7.14	7.16
V(B)	4014	2495	2996	3469	3924	8200	7491	7676	9246	9431
Ecart	3	1	1	1	3	1	4	6	4	6

Tableau 30

b) Le tableau 31 (resp. 32) fournit pour les problèmes de 50, 100, 500 et 1000 variables tirés au hasard (resp. pour chacune des séries des problèmes de l'Ecole des Mines du Colorado)

- la taille n des problèmes

- le nombre NP de problèmes

- pour chaque itération ITER j(j=1,....,4) de notre algorithme et pour la méthode de Laurière (colonnes Laurière) dans la partie NVR (resp. MAX) le nombre moyen (resp. maximum) de variables non éliminées en fin d'itération, ou de méthode pour Laurière

- en colonne ITER 1/DOM, la moyenne des pourcentages de variables éliminées directement par les relations de dominance (sans avoir à calculer les coût réduits associés)

n	N P	ITER 1			LAURIERE		ITER 2		ITER 3		ITER'
		NVR	MAX	DOM	NVR	MAX	NVR	MAX	NVR	MAX	NVR
50	100	11.20	48	84	4.22	19	2.87	19	2.01	13	0.72
100	100	11.24	35	89	5.53	15	3.90	15	2.90	9	1.12
500	30	17.67	76	97	6.83	20	5.57	20	3.40	17	1.13
1000	10	25.90	81	98	3.60	15	3.40	15	3.40	15	0

Tableau 31

Série	n	NP	ITER 1			LAURIERE		ITER 2		ITER 3		ITER 4	
			NVR	MAX	DOM	NVR	MAX	NVR	MAX	NVR	MAX	NVR	MAX
1	200	10	17.70	36	93.18	4.40	17	3.70	17	3.30	16	3	14
2*	100	10	12.44	25	88.18	5.44	13	3.56	13	2.11	10	1.11	10
3	100	10	26.70	51	82.60	6.40	20	4.80	20	2.90	15	1.90	13
4	100	10	29.70	60	83.09	9	25	8.70	25	6.70	22	4.80	18
5	100	10	16.20	35	88.26	5.50	9	4.50	9	3.40	8	1.30	8
6*	150	12	45.82	122	86.70	16.91	98	6.54	22	4.55	19	3.18	18
7	75	22	32.04	67	77.91	19.27	59	7.09	47	5.59	46	1.55	12

* moyenne effectuée sur NP-1 problèmes (la solution optimale d'une des NP relaxations est bivalente)

Tableau 32

c) Une comparaison entre les résultats obtenus pour les problèmes tirés au hasard et ceux obtenus pour les problèmes de l'Ecole des Mines du Colorado permet de conclure que pour ces problèmes :

1 - l'efficacite de l'itération 2 joue un rôle beaucoup plus important que pour les problèmes tirés au hasard

2 - l'écart entre les performances de notre algorithme de réduction et celles de la méthode de Laurière est encore accentué

3 - l'apport des itérations 3 et 4 n'est pas négligeable ; le tableau 33 en apporte la preuve ; il contient pour ITER j(j = 3 et 4) ; en colonne :

- NP le nombre de problèmes dont les variables ne sont pas toutes éliminées en fin d'itération ITER (j-1)

- MOY (resp. MAX) : le nombre moyen (resp. maximum) de variables éliminées en cours d'itération j.

Note

ITER 3(resp. 4) n'est d'aucune utilité pour 6 (resp. 8) des 47 (resp. 40) problèmes concernés.

□

	NP	Variables Eliminées	
		MOY	MAX
ITER 3	47	2.62	10
ITER 4	40	4.12	46

Tableau 33

VII.3 - ALGORITHME MSB 79

Une analyse très détaillée des résultats de notre méthode - dite MSB 71 ([K8, K10] et [G54])- destinée à mettre en relief l'importance des tests du tri des données, précède la donnée des résultats relatifs à la nouvelle version de notre algorithme - appelé MSB 79 - qui est prouvé être plus performant que celui de Martello et Toth publié tout récemment (1978) (voir [K35]) ; cette nouvelle version se décompose comme suit :

1 Résolution de ($\bar{B}$) par *l'algorithme NKR* (partie II)

2 *Réduction* de la taille de (B) (partie V) avec uniquement l'utilisation de *l'itération 1*

3 *Enumération implicite* (partie VI) avec une hiérarchie de variables basée sur une adaptation de la méthode de tri rapide avec "seuil" dans laquelle les sous-fichiers, obtenus en cours d'algorithme, de dimensions ne dépassant pas le seuil sont laissés en état (abandon du tri par insertion) (§ VII 1.1).

VII.3.1 - Analyse des résultats de la méthode MSB 71

Pour les problèmes du knapsack tirés au hasard suivant la loi uniforme, la progression des temps de calcul de la méthode MSB 71 (publiés dans [K8, K9, K10])en fonction du nombre n de variables est de l'ordre de n^2

Ces temps de calcul incluent les temps du classement des données suivant l'ordre décroissant des $c_j/a_j (j = 1,\ldots,n)$, qui constitue la phase initiale de l'algorithme CKR. L'étude détaillée des temps de résolution va permettre de mettre en lumière l'extrême importance des temps consacrés à ce tri, les autres temps (concernant la réduction et l'exploration) devenant négligeables à partir de 1000 variables.

Le tableau 34 a rassemble pour chacune des trois étapes

1 Relaxation (CKR)
2 Réduction
3 Exploration

- dans la colonne "temps" : les moyennes des temps de calcul

- dans la colonne "%" : les moyennes des poucentages des temps de calcul par rapport aux temps globaux de résolution.

Ces résultats mettent en lumière les points suivants :

(i) La méthode de tri utilisée dans MSB 71 (CKR Select) est de complexité $O(n^2)$.

(ii) Le pourcentage de ces temps par rapport aux temps globaux de résolution est une fonction croissante du nombre de variables (de 80 % pour 50 variables à 99,5 % pour 2000 variables).

(iii) La réduction et l'exploration sont de complexité $O(n)$ et occupent une place de plus en plus négligeable lorsque n croît (de 20 % pour 50 variables à 0,5 % pour 2000 variables).

Ainsi, si on ne tenait compte que de la réduction et de l'exploration, c'est-à-dire si on supposait les variables triées, la méthode MSB 71 serait de complexité $O(n)$ pour les problèmes tirés au hasard.

n	NP	Relaxation (CKR) Temps	%	Réduction Temps	%	Exploration Temps	%	Total
50	50	14.96	78.80	1.64	8.6	2.38	12.60	18.98
100	50	51.85	85.60	3.33	5.5	5.42	8.90	60.60
500	50	1135.79	97.40	12.77	1.1	17.08	1.50	1165.64
1000	50	4451.53	98.80	22.94	0.51	28.04	0.69	4502.51
2000	50	16500	99.41	45.09	0.27	52.71	0.32	16598

Tableau 34 a

Le tableau 34 b met en relief encore davantage la nécessité de définir une règle du jeu :

Les temps de résolution du knapsack par les méthodes classiques doivent-ils inclure ceux du tri des données ?

Si la réponse est positive - et c'est notre conviction profonde - que cet exposé démontre qu'il est possible de se passer du classement pour résoudre la relaxation du knapsack et d'accélérer ainsi sensiblement tous les algorithmes classiques de résolution du knapsack - dans la rubrique "AVEC TRI", la colonne "GH" (resp. "MSB 71) rappelle pour la méthode de Greenberg et Hegerich (resp. notre méthode) les moyennes des temps de calcul globaux répertoriés dans [K8], qui incluent les temps du tri des données (expériences réalisées sur *Gamma M40*, tous les temps sont exprimés en secondes).

Si la réponse est négative, sous le libellé "SANS TRI", les colonnes "G H" (méthode de Greenberg et Hegerich) et "MSB" (notre méthode) fournissent les moyennes des temps de calcul obtenus en excluant les temps nécessaires au tri des données.

Les résultats des colonnes

"GH/MSB 71" : rapports des temps de la méthode de Greenberg et Hegerich (T_{GH}) sur ceux de la méthode MSB (T_{MSB})

"GAIN" : pourcentage de $T_{GH} - T_{MSB}$ par rapport à T_{GH}

indiquent des conclusions tout-à-fait disproportionnées suivant la règle du jeu adoptée.

Ainsi, si l'on ne se réfère qu'à la rubrique "SANS TRI", la méthode MSB 71 est environ 10 fois (resp. 12 fois) plus rapide que celle de Greenberg et Hegerich pour les problèmes de 500 (resp. 1000) variables, le gain étant toujours de l'ordre de 90 % quelque soit la taille des problèmes.

	AVEC TRI				SANS TRI			
	GH	MSB 71	GH/ MSB71	Gain	GH	MSB 71	GH/ MSB71	Gain
50	1.80	1.26	1.43	30	.792	.252	3.14	91.97
100	6.44	4.26	1.51	33.85	2.627	.447	5.88	92.40
500	110.88	93.70	1.18	15.49	19.148	1.968	9.73	89.72
1000	409.65	368.85	1.11	9.95	44.489	3.689	12.06	91.71

Tableau 34 b

L'utilisation de la méthode de tri Quicksort conduit à une complexité pratique O(n) pour la méthode MSB 71 (voir tableau 35).

n	Alg CKR		Réduction	Exploration		Temps globaux		Gain
	Select	Quicks		Select	Quicks	Select	Quicks	
50	14.96	5.23	1.64	2.38	2.76	18.98	9.63	49.26
100	51.85	11.50	3.33	5.42	6.04	60.60	20.87	65.56
500	1135.79	73.07	12.77	17.08	16.51	1165.64	102.35	91.22
1000	4451.53	159.42	22.94	28.04	25.06	4502.51	207.42	95.39
2000	16500*	345.87	45.09	52.71	44.32	16598*	435.28	97.38

* uniquement pour 3 problèmes tests.

Tableau 35

VII.3.2 - Résultats de la méthode MSB 79

Les temps d'exécution étape par étape de la méthode MSB 79 (tableau 36) précèdent une comparaison des méthodes MSB 71 [avec utilisation du "tri rapide"], MSB 79 et celle de Martello et Toth. (tableau 37)

n	Relaxation	Réduction	Exploration	Total	Gain % MSB 71
50	2.99	1.23	2.39	6.61	45.69
100	5.62	2.37	5.22	13.21	36.70
500	27.91	10.63	11.46	50.00	51.15
1000	50.72	23.56	9.57	83.85	59.57
2000	112.15	31.75	21.19	165.09	62.07
5000	281.04	81.07	28.76	390.87	
7500	394.32	105.50	33.74	533.56	

Tableau 36

	DONNEES TRIEES			DONNEES NON TRIEES		
n	MSB71 avec Quick Sort	MSB 79	M.T.	MSB 71	MSB 79	M.T.
50	4.90	4.12	5.58	9.63	6.61	10.31
100	10.19	8.41	11.32	20.87	13.21	22.0
500	32.67	25.48	36.80	102.35	50.00	110.67
1000	55.14	40.27	51.48	207.42	83.85	203.76
2000	103.95	67.48	84.76	435.28	165.09	416.09
5000		143.36	183.10		390.87	1108.98
7500					533.56	

Tableau 37

Le tableau 38 montre (pour n = 1000) la faible influence de la valeur du second membre sur les temps de calcul.

L'intérêt du classement systématique, par ordre décroissant des pénalités, est inversement proportionnel à la taille du problème : lorsque n croît, de nombreuses pénalités étant du même ordre de grandeur, il est préférable d'employer la méthode de tri rapide avec seuil, dans laquelle les sous-fichiers obtenus en cours d'algorithme, de dimensions ne dépassant pas le seuil, sont laissés en état.

Le tableau 39 montre que le seuil optimal est de 35 (resp. 60) pour 5000 (resp. 7500) variables.

		MSB 79				M.T.
		Relaxation	Réduction	Exploration	Total	
$b = \alpha \Sigma a_j$	$\alpha = 0.2$	56.22	25.90	13.42	95.54	205.67
	$\alpha = 0.5$	55.33	25.02	14.23	94.58	209.18
	$\alpha = 0.8$	50.10	23.57	8.35	82.02	219.93
$\max_j a_j \leqslant b < \Sigma a_j$		50.72	23.56	9.57	83.85	203.76

Tableau 38

Seuil	5000 VARIABLES			7500 VARIABLES		
	Hiérarchie des variables	Exploration implicite	Temps global exploration	Hiérarchie des variables	Exploration implicite	Temps global exploration
0	24.04	32	56.04	29.03	21.36	50.39
8	21.32	22.31	43.63			
25	19.13	14.73	33.86			
35	18.84	9.92	28.76	24.42	12.19	36.61
50	18.40	11.03	29.43	23.77	10.71	34.48
60	18.14	13.14	31.28	23.41	10.33	33.74
75				23.07	11.67	34.74

Tableau 39

PROBLEMES EN NOMBRES

ENTIERS A PLUSIEURS CONTRAINTES

CHAPITRE 3

INTRODUCTION

Les points fondamentaux de la méthode proposée pour la résolution d'un problème à m contraintes et n variables discrètes du type

$$(B)\quad \begin{cases} \max\ cx \\ \text{s.c. } Ax = a \\ \quad x_j \in \{0 \text{ ou } 1\} \quad \forall j \in J_1 \\ \quad x_j \in \mathbb{N} \quad \forall j \in J_2 \\ \text{où } J_1 \cup J_2 = \{1,\ldots,n\} \text{ et } J_1 \cap J_2 = \emptyset \end{cases}$$

sont les suivantes :

1° *Réduction de la taille* de (B) à la fois en nombre de variables et en nombre de contraintes (voir partie I).

2° *Définition d'un schéma d'exploration* : de l'ensemble des points entiers du domaine : à partir d'un sommet de V ou génère les $2^n - 1$ autres sommets suivant un procédé du type lexicographique.

3° *Choix d'une partition de* $\{1,\ldots,n\}$:

Une partition $\{I, \bar{I}\}$ de $\{1,\ldots,n\}$ est déterminée pour appliquer le principe d'exploration sur le sous-ensemble de variables d'indices appartenant à $\bar{I}$ et résoudre à chaque étape un problème du type :

$$(PA)\quad \begin{cases} c_{\bar{I}} x_{\bar{I}} + \max\ c_I x_I \\ \quad \text{s.c. } A^I x_I = a - A^{\bar{I}} x_{\bar{I}} \\ \quad\quad x_j = 0 \text{ ou } 1 \quad \forall j \in I \end{cases} \qquad (45)$$

L'ensemble I est choisi de telle manière que l'exploration des solutions de (45) soit la plus simple possible compte-tenu de la structure de A. Différentes méthodes de résolution de (45) peuvent être employées ; nous en avons testées quatre :

3.1. Emploi d'une technique basée sur la réduite de Smith qui permet de se ramener à une matrice triangulaire inférieure infiniment plus maniable en pratique (voir [G15]).

3.2. Utilisation de la base optimale de $(\bar{B})$: I(resp. $\bar{I}$) désigne l'ensemble des indices de base (resp. hors-base) à l'optimum de $(\bar{B})$.

Le système du type

$$A^I x_I = a - A^{\bar{I}} x_{\bar{I}}$$

peut être résolu de deux manières :

a) Emploi du tableau simplicial à l'optimum de $(\bar{B})$ résolu par la méthode simpliciale écrite en rationnel, pour considérer :

$$x_I = (A^I)^{-1} a - (A^I)^{-1} A^{\bar{I}} x_{\bar{I}} \text{ avec } (A^I)^{-1} a_i \text{ , } (A^I)^{-1} a_{ij} \in \mathbb{Q} \ \forall i, j$$

b) Utilisation de

$\det(A^I)$: déterminant de A^I

B^I : matrice des cofacteurs de A^I

pour considérer $\det(A^I)\, x_I = B^I a - B^I A^{\bar{I}} x_{\bar{I}}$.

Ces procédés qui, en théorie, permettent de considérer des problèmes de tailles quelconques, sont pratiquement très vite limités par la grandeur des entiers obtenus ou par la durée du temps de calcul.

3.3 Transformation en un problème du knapsack équivalent par contraction d'un sous-ensemble de contraintes en une contrainte unique dont l'ensemble des solutions entières réalisables est identique à celui du système (voir partie II).

3.4. Choix d'un sous-ensemble I tel que la matrice A^I soit diagonale (lorsque la structure de A le permet) :

Cette méthode a été utilisée avec succès lors de la résolution du problème de fabrication exposée en partie III.

I

REDUCTION POUR LES PROBLEMES A PLUSIEURS CONTRAINTES

INTRODUCTION

Le but de cette partie est de montrer que la réduction (à la fois en nombre de variables et en nombre de contraintes) d'un problème linéaire en variables 0-1 peut être réalisée efficacement par la conjonction de tests simples appliqués, en particulier, sur des problèmes du knapsack extraits du problème d'origine.

L'énoncé des tests d'élimination définitive de variables et de contraintes (§ I.2) précède la présentation de notre algorithme de réduction (§ I.3) (voir [G49]).

Ces travaux, entrepris en collaboration avec Pierre Hansen, ont abouti à des résultats numériques satisfaisants (§ I.4), les expériences ayant été réalisées sur CII 10070 à partir de problèmes d'origine concrète donnés par Petersen [G48].

I.1 - FORMULATION DU PROBLÈME :

L'étude de la réduction des problèmes linéaires à m contraintes et n variables bivalentes

$$(B) \quad \begin{cases} \max \ cx \\ \text{s.c. } Ax \leq b \\ \qquad x \in V \end{cases}$$

[dont les coefficients c_j de c, b_i de b et a_{ij} de A ($i = 1,\ldots,m$; $j = 1,\ldots,n$) sont supposés être des entiers non négatifs (caractéristique que possèdent les problèmes de Petersen)] aura pour support l'utilisation de problèmes du knapsack extraits de (B) notés comme suit, pour k, p $\in \{1,\ldots,m\}$ donnés :

$$(B_p^0) \quad \begin{cases} \max \ cx \\ \text{s.c. } A_p x \leq b_p \\ \qquad x \in V \end{cases} \quad ; \quad (B_p^k) \quad \begin{cases} \max \ A_k x \\ \text{s.c. } A_p x \leq b_p \\ \qquad x \in V \end{cases}$$

I.2 - TESTS DE RÉDUCTION DE LA TAILLE DE (B)

Les tests exposés ci-dessous sont énoncés avec les problèmes (B), (B_p^0), (B_p^k) d'origine ; ce qui permet l'élimination *définitive* de variables et de contraintes.

Il va de soi que si ces tests étaient utilisés en cours d'itération d'une méthode d'exploration, donc avec un sous-ensemble de variables fixées à priori :

$$x_j = \bar{x}_j \in \{0,1\} \quad j \in W \subset \{1,\ldots,n\} \quad (H)$$

les éliminations déduites des tests ne seraient valables que sous l'hypothèse (H). De plus, il y aurait lieu d'ajouter le résultat suivant :

étant donné (SB) le sous-problème de (B) pour lequel (H) est supposée vérifiée, si l'une au moins des valeurs suivantes

$$\text{(i)} \quad \lfloor v(\overline{SB}) \rfloor$$
$$\text{(ii)} \quad \lfloor v(\overline{SB}_p^0) \rfloor$$
$$p \in \{1,\ldots,m\},\ u \in \mathbb{R}_+$$
$$\text{(iii)} \quad \lfloor v(RSB_p^0)(u) \rfloor$$

est inférieure à $\underline{v}(B)$ alors il est inutile de considérer le sous-ensemble $\{x \in V \mid x_j = \bar{x}_j,\ j \in W\}$.

I.2.1 - Elimination de variables

Théorème 28

Etant donnés

$k \in \{1,\ldots,m\}$, $\varepsilon \in \{0,1\}$,
$u \geq 0$ de *dimension convenable*,

s'il existe $j \in \{1,\ldots,n\}$ *tel que*

(V1) $a_{kj} > b_k$

alors la variable d'indice j doit être fixée à la valeur 0 ;

(V2) $\lfloor v(RB_k^o(u) \mid x_j = \varepsilon) \rfloor \leq \underline{v}(B)$

ou

(V3) $\lfloor v(\bar{B}_k^o \mid x_j = \varepsilon) \rfloor \leq \underline{v}(B)$

ou

(V4) $\lfloor \max \{v(\bar{B}_k^o \mid x_j = \varepsilon,\ x_{i(\varepsilon)} = 0),\ v(\bar{B}_k^o \mid x_j = \varepsilon,\ x_{i(\varepsilon)} = 1)\} \rfloor \leq \underline{v}(B)$

avec $x_{i(\varepsilon)}$ *variable de base optimale de* $(\bar{B}_k^o \mid x_j = \varepsilon)$

ou

(V5) $\lfloor v(RB(u) \mid x_j = \varepsilon) \rfloor \leq \underline{v}(B)$

alors la variable d'indice j doit être fixée à la valeur $1 - \varepsilon$.

Démonstration : Le résultat relatif à (V1) est évident ; les autres résultats découlent de ce que les différentes quantités envisagées sont des majorants de $v(B \mid x_j = \varepsilon)$.

□

Etant donnés

$k \in \{1,\ldots,m\}$

$i(k)$ indice de base optimale de $(\bar{B}_k^o)$

on considère les problèmes

$$(\bar{B}_k^o \mid x_{i(k)} = 0) \text{ et } (\bar{B}_k^o \mid x_{i(k)} = 1).$$

L'application des tests (V2) et/ou (V3) et/ou (V4) permet la construction des ensembles

$X_{\delta,\varepsilon} = \{j \in \{1,\ldots,n\} \mid x_j$ fixée à la valeur ε lorsque $x_{i(k)} = \delta\}$ $\varepsilon, \delta \in \{0,1\}$

pour énoncé le

Théorème 29 (*voir* [K11])

Etant donnés $k \in \{1,\ldots,m\}$, $\varepsilon \in \{0,1\}$, *si*

$$(V6)\quad X_\varepsilon = X_{0,\varepsilon} \cap X_{1,\varepsilon} \neq \emptyset$$

alors les variables d'indices $j \in X_\varepsilon$ *doivent être fixées à la valeur* ε.

Démonstration : Toute variable d'indice $j \in X_\varepsilon$ (supposé non vide) doit prendre la même valeur sous les hypothèses complémentaires $x_{i(k)} = 0$ et $x_{i(k)} = 1$.

□

I.2.2 - Elimination de contraintes

Théorème 30

La contrainte

$$A_k\, x \leq b_k \qquad k \in \{1,\ldots,m\}$$

peut être éliminée du problème (B) *sous l'une des hypothèses suivantes :*

(C1) $\sum_{j=1}^{n} a_{kj} \leq b_k$

$\exists p \in \{1,\ldots,k-1,k+1,\ldots,m\}$ *tel que*

(C2) $a_{kj} \leq a_{pj} \quad j = 1,\ldots,n$

$b_k \geq b_p$

(C3) $\lfloor v(\bar{B}_p^k) \rfloor \leq b_k$

(C4) $q_p = \lfloor \max \{v(\bar{B}_p^k \mid x_{i(p)} = 0),\ v(\bar{B}_p^k \mid x_{i(p)} = 1)\} \rfloor \leq b_k$

avec i(p) *indice de base optimale de* $(\bar{B}_p^k)$.

$\exists \varepsilon \in \{0,1\}: q_p > b_k$ *et* $\lfloor v(\bar{B}_p^k \mid x_{i(p)} = 1-\varepsilon) \rfloor \leq b_k$

(C5) $q_{p,\varepsilon} = \lfloor \max\{v(\bar{B}_p^k \mid x_{i(p)}=\varepsilon, x_{i(p,\varepsilon)}=0), v(\bar{B}_p^k \mid x_{i(p)}=\varepsilon, x_{i(p,\varepsilon)}=1)\} \rfloor \leq b_k$

avec i(p,ε) *indice de base optimale de* $(\bar{B}_p^k \mid x_{i(p)} = \varepsilon)$.

Démonstration. :

(C1) évidente

(C2) $\Rightarrow \{x \in V \mid A_p x \leq b_p\} \subset \{x \in V \mid A_k x \leq b_k\}$

(C3) $\Rightarrow \forall x \in V : A_p x \leq b_p \Rightarrow A_k x \leq \lfloor v(\bar{B}_p^k) \rfloor \leq b_k$

(C4) et (C5) aboutissent par séparations à des bornes supérieures de $v(B_p^k)$ inférieures à b_k. □

Notes

(i) (C2) est un cas particulier du point (C3) dû à Hansen [K26]

(ii) dans le cadre d'une méthode d'exploration, si l'hypothèse précédant le test (C5) est vérifiée mais si $q_{p,\varepsilon} > b_k$ alors il est possible de conclure que la contrainte k peut être éliminée sous l'hypothèse $x_{i(p)} = 1-\varepsilon$.

□

I.3 - ALGORITHME DE RÉDUCTION

L'algorithme de réduction suivant donne un aperçu de l'utilisation des tests d'élimination de variables et de contraintes exposés ci-dessus (d'autres façons d'enchainer ces tests pourraient bien entendu, être envisagées).

En posant

X_0 (resp. X_1) l'ensemble d'indices des variables éliminées à 0 (resp. 1)

E l'ensemble d'indices des contraintes éliminées

$X_2 = \{1,\ldots,n\}\setminus(X_0 \cup X_1)$ et $\bar{E} = \{1,\ldots,m\}\setminus E$

les différents tests seront appliqués, pour X_0, X_1, E donnés, au problème

$$\left[\begin{array}{l} \sum_{j\in X_1} c_j + \max \sum_{j\in X_2} c_j x_j \\ \qquad \text{s.c.} \sum_{j\in X_2} a_{ij} x_j \le b_i - \sum_{j\in X_1} a_{ij} \qquad \forall i \in \bar{E} \\ \qquad\quad x_j = 0 \text{ ou } 1 \qquad\qquad \forall j \in X_2 \end{array}\right.$$

Principe :

Phase 1 :

Elimination évidente (tests (C1) et (V1)) de contraintes et de variables.

Phase 2 :

"Préparation" de l'élimination de variables :

Calcul des valeurs des relaxations de chacun des knapsacks (B_k^0) $k=1,\ldots,m$ extraits de (B). Classement des contraintes de telle sorte que :

$v(\bar{B}_1^0) \le v(\bar{B}_2^0) \le \ldots \le v(\bar{B}_m^0)$.

Phase 3 :

Elimination de variables par l'intermédiaire d'algorithmes de réduction appliqués tour-à-tour sur (B_1^o), (B_2^o),..., (B_m^o).

Phase 4 :

Elimination de contraintes :

Tests appliqués sur (B_1^k) k = 2,...,m.

Phase 5 :

Elimination de variables en utilisant :

- les contraintes composites

- la relaxation du problème (B) réduit :

Algorithme détaillé

0 $\underline{v}(B)$, $\mu \in]0,1[$ donnés ; $X_0 \equiv X_1 \equiv E \equiv \emptyset$

1 1.1 $n \leftarrow |X_2|$, $k \leftarrow 1 \in \bar{E}$

2 1.2 $m \leftarrow |\bar{E}|$

Si (C1) vrai pour contrainte k *alors* $E \equiv E \cup \{k\}$

3 *aller en* 1.2

Si (V1) vrai pour variable x_j, $j \in X_2$ et la contrainte k *alors* $X_0 \equiv X_0 \cup \{j\}$

aller en 1.1

4 calcul de $v(\bar{B}_k^o)$

si k < m *alors* k←k+1

aller en 2

5 classement des contraintes* suivant l'ordre croissant des valeurs des knapsacks $(\bar{B}_k^o)$; on suppose donc que

$v(\bar{B}_1^o) \leq v(\bar{B}_2^o) \leq \ldots \leq v(\bar{B}_m^o)$

$k \leftarrow 1$

6 i(k) indice de base optimale de $(\bar{B}_k^o)$

pour j = 1, 2,...,i(k) - 1, i(k) + 1,...n faire
$\varepsilon \in \{0,1\}$ donné
si (V2) vrai pour x_j alors aller en 6.1
si $\lfloor v(RB_k^o(u) \mid x_j = \varepsilon) \rfloor \leq \mu . v(B)$ alors
si (V3) vrai pour x_j alors aller en
si (V4) vrai pour x_j alors aller en
sinon aller en 6.2

6.1 $X_\varepsilon \equiv X_\varepsilon \cup \{j\}$

6.2 Stop

si $\exists \varepsilon \in \{0,1\}$:(V3) vrai pour $x_{i(k)}$ alors $X_\varepsilon \equiv X_\varepsilon \cup \{i(k)\}$

application de (V6) $\Rightarrow$ modification possible de X_ε, $\varepsilon \in \{0,1\}$

Si $\exists \varepsilon \in \{0,1\}$:$X_\varepsilon$ a été modifié à l'étape k alors aller en 1.1**
sinon si k < m alors $k \leftarrow k+1$
aller en

7 pour k = 2,...,m faire
Si (C3), puis (C4), puis (C5) vrai pour $(\bar{B}_1^k)$ alors $E \equiv E \cup$
Si $\exists p \in \{1,\ldots,k-1,k+1,\ldots,m\}$:(C3) vrai pour (k,p) alors $E \equiv E \cup$

8 pour j = 1,...,n faire
si $\exists \varepsilon \in \{0,1\}$:(V5) vrai pour x_j alors $X_\varepsilon \equiv X_\varepsilon \cup \{j\}$

si $\exists \varepsilon \in \{0,1\}$:$X_\varepsilon$ a été modifié alors aller en 1.1
sinon stop.

* le classement peut être également basé sur les valeurs obtenues par séparation, i.e. max $\{v(\bar{B}_k^o \mid x_{i(k)} = 0), v(\bar{B}_k^o \mid x_{i(k)} = 1)\}$, $k \in \bar{E}$.

** le retour à l'étape 1 est inutile lorsque les variables sont fixées à leurs valeurs optimales dans chacun des knapsacks $(\bar{B}_k^o)$, $k \in \bar{E}$.

*** ce point n'est à envisager que si m vaut quelques unités.

□

I.4 - EXPÉRIENCES NUMÉRIQUES

L'algorithme de réduction a été implanté sur CII 10070 pour considérer les 7 problèmes donnés par Petersen dans [G48]

Pour chacun d'eux, les valeurs expérimentales de u ont été les multiplicateurs optimaux associés à $(\bar{B}_k^o)$ $k \in \bar{E}$ pour (V2), $(\bar{B})$ pour (V5). Ainsi, on a considéré à chaque fois les meilleures relaxations lagrangiennes au sens des relations (43) et (44) qui ne correspondent pas nécessairement au plus grand nombre de variables éliminées (voir § IV.3 du chapitre 2). Mais l'utilisation du test (V6), pour les problèmes $(\bar{B}_k^o)$ $k \in \bar{E}$, a permis de tenir compte d'autres relaxations lagrangiennes.

Le tableau 40 indique, pour chacun des 7 problèmes, en colonne

- *contraintes* (resp. *variables*) : les nombres de contraintes (resp. variables) du problème initial (I) et réduit (R) après utilisation de l'algorithme.

- $v(B)$: la valeur optimale du problème
- $\lfloor v(\bar{B}_k^o) \rfloor$: les parties entières des valeurs des 3 "meilleurs" knapsacks et les valeurs correspondantes de k.
- $\lfloor v(\bar{B}) \rfloor$: la partie entière de la valeur de la relaxation du problème initial, le nombre de variables éliminées (V5-I) par application directe du test (V5).

. Ci (i = 1,...,5) (resp. Vi(i = 1,...,6)) : le nombre de contraintes (resp. variables) éliminées par le test associé.

. K : les indices k des knapsacks (B_k^o) pour lesquels au moins un des tests (Vi) i = 1, 2, 3, 4, 6 a été efficace.

Le tableau 41 précise :

- les indices des variables fixées définitivement à 0(colonne X_0) et à 1 (colonne X_1).
- les numéros des contraintes éliminées (colonne E).

Notes

(i) Pour chaque problème, on a posé $\underline{v}(B) = v(B)$, valeur donnée dans [G49] . (Se référer à [G3, G27, G60] pour des descriptions d'heuristiques).

(ii) problèmes 1 et 2 : ils se ramènent chacun à un problème du knapsack dont toutes les variables sont finalement éliminées ce qui prouve l'optimalité de la valeur $\underline{v}(B)$.

(iii) problème 2 : ce sont les tests appliqués au knapsack (B_2^o) $(v(\bar{B}_2^o) = \min_{1\leq k\leq 10} v(\bar{B}_k^o))$ qui permettent l'élimination du plus grand nombre de variables. Mais une variable supplémentaire est éliminée grâce à l'application du test (V4) sur le knapsack (B_3^o) qui est tel que $v(\bar{B}_3^o) > v(\bar{B}_k^o)$ $\forall k \neq 7$ ...

(iv) Déroulement de l'élimination des variables :

Problème 4 : 11 variables éliminées avec (B_2^o), puis 1 avec (B_{10}^o), puis 1 avec le test (V5), et enfin 1 avec (B_2^o).

Problème 5 : 15 variables éliminées avec (B_2^o), puis 2 avec (B_{10}^o) et enfin 1 avec le test (V5).

(v) Problèmes 6 et 7 : l'échec des tests (Vi) $i \neq 5$ appliqués aux knapsacks du type (B_k°) conduit à l'étude de la construction de problèmes du knapsack dont les contraintes sont combinaison linéaire des contraintes d'origine.

A titre d'information :

Pour le problème 6 : la somme des contraintes 1, 2 et 5 aboutit à un knapsack noté (B_{1+2+5}°) tel que $\lfloor v(\bar{B}_{1+2+5}^\circ) \rfloor = 10760$ et avec lequel 7 variables sont éliminées.

Pour le problème 7 : le knapsack (B_{1+5}°) tel que $\lfloor v(\bar{B}_{1+5}^\circ) \rfloor = 16761$ permet l'élimination de 11 variables.

□

Problème	Contraintes		Variables		v(B)	$\lfloor v(\bar{B}_k^o) \rfloor$		$\lfloor v(\bar{B}) \rfloor$		C1	C2	C3	C5	V1	V2	V3	V4	V5	V6	K
	I	R	I	R			$\frac{k}{n}$	V5.I													
1	10	1	6	0	3800	4271 4325 4400	5 1 6	4134	2	9				2	3		1			1,5
2	10	1	10	0	87061	93780 99462 102145	2 10 8	92977	2	9				3	3	3	1			2,3
3	10	3	15	8	4018	4149 4211 4233	10 9 8	4127	3			6	1	1	4	1			1	10,8
4	10	2	20	6	6120	6184 6540 6555	2 10 1	6155	12	2		6			10	1	1	1	1	2,10
5	10	2	28	10	12400	12523 13022 13312	2 10 1	12462	15	2	1	5			14		1	1	2	2,10
6	5	4	39	28	10618	11122 11444 11536	1 2 3	10668	11			1			1			10		1
7	5	4	50	35	16537	17071 17079 17999	1 2 5	16601	15			1			3			12		1

Tableau 40

Problème	X_0	X_1	E
1	Toutes les variables sont éliminées		1,2,3,4,6,7,8,9,10
2			1,3,4,5,6,7,8,9,10
3	11, 13	2,6,10,14,15	1,2,3,4,5,6,9
4	4,5,6,7,8,10,11,12,13	1,14,15,17,20	1,3,4,5,6,8,9,10
5	4,5,7,8,11,12,13	14,15,17,20,21,22, 23,25,26,27,28	1,3,4,5,6,7,8,9
6	3,5	4,6,11,15,16,17, 27,29,32	4
7	3,5,46	6,11,15,16,17,32, 40,41,42,43,47,48	4

Tableau 41

II

RESOLUTION D'UN PROBLEME D'AFFECTATION

INTRODUCTION

L'objet de cette étude est la résolution d'un problème d'origine concrète d'implantations de groupes de divers sites sur plusieurs périodes (énoncé et formulé au § II.1) dont la particularité est de posséder des contraintes du type $\sum_j x_j = 1$ (problème de partitionnement), l'ensemble S des autres contraintes n'admettant pour éléments non nuls que des coefficients égaux à 1.

Ces caractéristiques associées à la bivalence des variables font que la contraction des contraintes de S aboutit à une équation unique équivalente dont les coefficients sont de grandeurs raisonnables (par exemple pour $m = 4$ et $n = 160$, la contrainte $ax = b$ résultante est telle que $\max_j a_j = 400$ et $b = 1600$, voir § II.3).

Cette contraction permet ainsi de générer un problème du knapsack équivalent, à variables bivalentes devant vérifier, outre la contrainte d'égalité, des conditions du type "partitionnement" - dite G.U.B (Generalized Upper Bound) - La méthode de résolution est une adaptation de celle exposée au chapitre 2 ; elle prend effectivement en compte au niveau des calculs, les contraintes "G.U.B" (§ II.2).

La première phase de notre méthode consistant en la résolution de la relaxation du problème, deux choix sont possibles : ou bien considérer le problème initial, ou bien considérer le problème obtenu après contraction.

Au § II.3, on verra, sur les exemples traités, que le premier choix est le plus judicieux : il permet d'aboutir à des temps de résolution comparables à ceux de [K8] qui étaient relatifs à des problèmes tirés au hasard.

II.1 - PROBLÈME ÉTUDIÉ

II.1.1 - Enoncé

Etant donné p emplacements, il s'agit d'installer au meilleur prix q moyens de production au cours d'un temps découpé en m périodes $t_i (i=1,\ldots,m)$ en tenant compte des contraintes suivantes :

1) Impossibilité d'installer, au cours d'une période, plus d'un moyen de production sur un emplacement donné. [Le nombre de moyens installés sur un emplacement est donc compris entre 0 et m].

2) Pour chaque période $t_i (i \in \{1,\ldots,m\})$, le nombre u_i de moyens de production à installer est imposé (de telle sorte que $\sum_{j=1}^{m} u_i = q$).

Ainsi, connaissant, pour chacun des emplacements, le coût d'installation de $r \in \{0,1,\ldots,m\}$ moyens de production (coût dépendant de r et de la chronologie des installations), il faut minimiser la somme globale des coûts d'installations.

II.1.2 - Formulation

D'après la condition 1, pour un emplacement et une période donnés, on peut installer ou non un moyen de production ; de plus, il faut tenir compte de la chronologie des installations. Ainsi, pour un emplacement donné cela implique l'existence de 2^m chroniques d'installation envisageables (de puis l'installation, au cours des m périodes d'aucun moyen de production jusqu'aux installations de m moyens de production (une au cours de chaque période)).

Ces 2^m chroniques peuvent être représentées par la matrice d'affectation $L(m \times 2^m)$ suivante, indicée en lignes par les périodes et en colonnes par les chroniques d'installation, dans laquelle pour une période t_i et une chronique j données, on a

$$\ell_{ij} = \begin{cases} 1 \text{ si un moyen de production est installé} \\ 0 \text{ sinon} \end{cases}$$

	1	2	3	4	5	...	2^{m-1}	$2^{m-1}+1$	$2^{m-1}+2$	$2^{m-1}+3$	$2^{m-1}+4$	$2^{m-1}+5$	...	2^m
1	0	0	0	0	0		0	1	1	1	1	1	...	1
2	0	0	0	0	0	...	1	0	0	0	0	0	...	1
.	.	.	.	.	.	...	.	.	.	.	.	.	...	.
m-3	0	0	0	0	0	...	1	0	0	0	0	0	...	1
m-2	0	0	0	0	1	...	1	0	0	0	0	1	...	1
m-1	0	0	1	1	0	...	1	0	0	1	1	0	...	1
m	0	1	0	1	0	...	1	0	1	0	1	0	...	1

Parmi ces 2^m chroniques, il ne faudra en retenir qu'une et une seule ; on va donc associer à chaque chronique j une variable d'état bivalente x_j telle que :

$$x_j = \begin{cases} 1 \text{ Si } j \text{ est adoptée} \\ 0 \text{ sinon} \end{cases}$$

Le choix d'une chronique impliquant l'élimination des $2^m - 1$ autres, il faut considérer la contrainte suivante (du type "G.U.B.")

$$\sum_{j=1}^{2^m} x_j = 1$$

Tout ce raisonnement étant valable pour chacun des p emplacements, on est conduit finalement à poser :

$J_k = \{(k-1)\,2^m + 1, \ldots, k\,2^m\}$ = ensemble des indices des 2^m chroniques attachées à un emplacement $k \in \{1,\ldots,p\}$.

f_{J_k} : vecteur ligne des coûts d'installations associés aux chroniques de l'emplacement $k \in \{1,\ldots,p\}$

Pour formuler le problème de la façon suivante :

$$(PB)\quad \left[\begin{array}{l} \min \sum_{k=1}^{p} \sum_{j\in J_k} f_j\, x_j \\ \text{s.c. } L_i\Big(\sum_{k=1}^{p} x_{J_k}\Big) = u_i \quad i = 1,\ldots,m \qquad (46) \\ \sum_{j\in J_k} x_j = 1 \qquad k = 1,\ldots,p \qquad (47) \\ x_j = 0 \text{ ou } 1 \quad \forall j \in \bigcup_{k=1}^{p} J_k \qquad (48) \end{array}\right.$$

avec $\forall i\ \forall j\ u_i \in \mathbb{N}$, $f_j \in \mathbb{Z}$, $\ell_{ij} = 0$ ou 1.

Note

Dans le cas particulier où $u_i = 1\ \forall i$, (PB) est un problème de partitionnement.

□

Si on pose :

$$\left[\begin{array}{l} n = p.2^m \text{ : nombre de variables de (PB)} \\ V^* = \{x \in V \mid \sum_{j\in J_k} x_j = 1,\ k = 1,\ldots,p\} \\ f = [f_{J_1}, f_{J_2},\ldots,f_{J_p}],\ \tilde{L}_i = \underbrace{[L_i, L_i,\ldots, L_i]}_{p \text{ fois}} \quad \forall i \\ x = \begin{bmatrix} x_{J_1} \\ x_{J_2} \\ . \\ . \\ . \\ x_{J_p} \end{bmatrix} \end{array}\right.$$

(PB) s'écrit :

$$\begin{cases} \min \ fx \\ \text{s.c. } \tilde{L}x = u \\ \qquad x \in V^* \end{cases}$$

x^* désignera une solution optimale de (PB).

II.1.3 - Propriétés relatives aux seconds membres des contraintes

Si pour une période donnée t_{i_o}, le nombre de moyens à installer est supérieur au nombre d'emplacements, alors le problème est impossible, c'est ce que traduit la

Propriété 4

Si $\exists i_o \in \{1,\ldots,m\}$ *tel que* $u_{i_o} > p$ *alors* $F(PB) = \emptyset$

Ainsi, pour un problème réalisable le nombre maximum de moyens installés sur p emplacements au cours des m périodes est-il égal à pm.

On pose

$$P_i = \{j \in \{1,\ldots,n\} \mid \tilde{\ell}_{ij} = 1\}$$

$$N_i = \{j \in \{1,\ldots,n\} \mid \tilde{\ell}_{ij} = 0\} \qquad i = 1,\ldots,m$$

Ces ensembles sont tels que :

$$\begin{cases} P_i \cup N_i = \bigcup_{k=1}^{p} J_k \\ P_i \cap N_i = \emptyset \qquad\qquad i = 1,\ldots,m \qquad (49) \\ |P_i| = |N_i| = \frac{n}{2} = p.2^{m-1} \qquad (50) \end{cases}$$

On a alors la

Propriété 5

$\forall x \in F(PB)$

(i) *si* $\exists i_o \in \{1,\ldots,m\}$ *tel que* $u_{i_o} = p$ *alors* $x_j = 0 \ \forall j \in N_{i_o}$ (51)

(ii) *si* $\exists i_o \in \{1,\ldots,m\}$ *tel que* $u_{i_o} = 0$ *alors* $x_j = 0 \ \forall j \in P_{i_o}$ (52)

Démonstration

$$\left.\begin{matrix}(51)\\ x \in F(PB)\end{matrix}\right\} \Rightarrow \tilde{L}_{i_o} x = p = \max\{\tilde{L}_{i_o} x \mid x \in V^*\} \underset{(47)}{\Rightarrow} \{j \mid x_j = 1\}$$

la relation (49) permet alors de conclure.
(raisonnement analogue pour la deuxième partie de cette propriété).

□

Propriété 6

Une contrainte $\tilde{L}_{i_o} x = u_{i_o}$ *telle que (51) (ou(52)) soit vérifiée peut être éliminée du problème (PB). Il en est de même pour les variables d'indices appartenant à* N_{i_o} *(ou* P_{i_o}*).*

Démonstration

Conséquence directe de la propriété 5, par définition de V^* et d'après (50).

□

D'où le

Corollaire

S'il exixte t seconds membres u_i *du type (51) ou (52), les t contraintes associées peuvent être éliminées du problème (PB) dont le nombre de variables est alors divisé par* 2^t.

D'après les propriétés précédentes, on supposera par la suite que le problème (PB) est tel que :

$$1 \leq u_i \leq p - 1 \quad i = 1,\ldots,m \tag{53}$$

II.2 - RÉSOLUTION DU PROBLÈME (PB)

II.2.1 - Modification de la formulation du problème

Un problème équivalent au problème (PB) est obtenu en contractant le système de contraintes (46) en une contrainte unique

$$\sum_{j=1}^{n} a_j x_j = b \tag{54}$$

dont l'ensemble des solutions bivalentes réalisables est identique à celui du système (46) :

$$F(PB) = \{x \in V^* \mid \sum_{j=1}^{n} a_j x_j = b\}.$$

En posant $c_j = -f_j (j = 1,\ldots,n)$, le problème équivalent à (PB) prend donc la forme suivante :

$$(B) \quad \begin{cases} \max \sum_{j=1}^{n} c_j x_j \\ \text{s.c.} \sum_{j=1}^{n} a_j x_j = b \\ x \in V^* \end{cases}$$

x^* désignera toujours une solution optimale de (B)
($v(B) = - v(PB)$).
et on notera $\bar{\bar{x}}$ (respectivement $\bar{x}$) une solution optimale de $(\bar{B})$ (resp. $(\overline{PB})$).

II.2.2 - Description de la méthode

Dans ce qui suit sont exposées les points fondamentaux de l'adaptation de la méthode décrite au chapitre 2.

II.2.2.1 - RESOLUTION DE LA RELAXATION

La première phase de la méthode consistant en la résolution de la relaxation du problème, il faut faire le choix entre $(\bar{B})$ et $(\overline{PB})$.

A priori, on pourrait choisir $(\bar{B})$ dont la structure particulièrement simple du système de contraintes permet d'établir les quelques propriétés suivantes débouchant sur une programmation simple et efficace :

a) Il est clair que

$$\left.\begin{array}{l} \sum_{j \in J_k} x_j = 1 \quad k = 1,\ldots,p \\ \\ x_j \geq 0 \quad j = 1,\ldots,n \end{array}\right\} \Rightarrow x_j \leq 1 \quad j = 1,\ldots,n$$

Ainsi la méthode simpliciale permet de résoudre le problème $(\bar{B})$ en ne prenant en compte que les conditions de non négativité des variables. (Cette remarque s'applique également pour $(\overline{PB})$.)

b) Ce problème comportant p + 1 contraintes, on sait que, pour une solution de base réalisable, les composantes hors-base sont toutes nulles et, seules les composantes de base (au nombre de p + 1) peuvent être positives non nulles.

On a alors la

Propriété 7

Une solution de base réalisable de $(\bar{B})$ possède nécessairement p - 1 ou p composantes égales à 1.

C'est une conséquence directe du lemme suivant pour lequel on pose :

$$\begin{cases} J = \{1,\ldots,n\} \\ \forall x \in \mathbb{R}^n \quad Q(x) = \{j \in J \mid 0 < x_j \leq 1\} \\ \qquad\qquad 1(x) = \{j \in Q(x) \mid x_j = 1\} \end{cases}$$

Lemme

Etant donnée une partition $(J_k)_{k=1,\ldots,p}$ *de l'ensemble* J, *une solution réalisable du système*

$$(T) \quad \begin{cases} \left.\begin{array}{l} \sum_{j \in J_k} x_j = 1 \quad k = 1,\ldots,p \end{array}\right\} \\ x_j \geq 0 \qquad \forall j \in J \\ |Q(x)| \leq p + 1 \end{cases} \tag{55}$$

possède l'une des deux formes suivantes

forme 1

$$\forall k \in \{1,\ldots,p\} \quad \exists i_k \in J_k \text{ tel que } \begin{cases} x_{i_k} = 1 \\ x_j = 0 \ \forall j \in J_k - \{i_k\} \end{cases}$$

forme 2

$$\exists k_o \in \{1,\ldots,p\} \quad \exists i_{k_1}, i_{k_2} \in J_{k_o} \text{ tel que } \begin{cases} x_{i_{k_1}} + x_{i_{k_2}} = 1 \\ x_j = 0 \ \forall j \in J_{k_o} - \{i_{k_1}, i_{k_2}\} \end{cases}$$

$$\forall k \in \{1,\ldots,k_o - 1,\ k_o + 1,\ldots,p\} \ \exists i_k \in J_k \text{ tel que } \begin{cases} x_{i_k} = 1 \\ x_j = 0 \ \forall j \in J_k - \{i_k\} \end{cases}$$

Démonstration

Par hypothèse, on a la relation $|Q(x)| \leq p + 1$ de plus,

$$(55) \Rightarrow |1(x)| \leq p \leq |Q(x)| \tag{56}$$

Si on démontre que $x \in \mathbb{R}^n$ ne peut être solution réalisable de (T) que dans l'une des deux hypothèses suivantes :

$$|Q(x)| = |\mathbf{1}(x)| = p \qquad \text{(H1)}$$
$$|Q(x)| = p + 1 \text{ et } |\mathbf{1}(x)| = p - 1 \qquad \text{(H2)}$$

le lemme sera démontré puisque, d'après (56) sous l'hypothèse (H1) (resp. (H2)) x ne peut être solution réalisable de (T) que si elle possède la forme 1 (resp. 2).

- Supposons que $|Q(x)| = p + 1$: deux cas sont envisagés :

cas 1 : $|\mathbf{1}(x)| = p \Rightarrow \exists k_o \in \{1,\dots,p\} \sum_{j \in J_{k_o}} x_j > 1 \Rightarrow$ (T) non vérifié

cas 2 : $|\mathbf{1}(x)| = p' \leq p - 2$: il est possible de saturer p' contraintes du système (55) : par exemple

$$\sum_{j \in j_k} x_j = 1 \quad k = 1,\dots,p'$$

mais il reste encore à satisfaire $p - p'$ contraintes avec seulement $|Q(x)| - |\mathbf{1}(x)| = p - p' + 1$ variables prenant des valeurs non nulles et inférieures à 1, ce qui est impossible.

- Supposons que $|Q(x)| = p$: si $|\mathbf{1}(x)| < p$, par un raisonnement analogue à celui du cas 2, on montrerait que x ne peut être solution réalisable de (T).

□

En conclusion, à l'optimum de $(\bar{B})$, deux cas peuvent se produire :

1. - $\bar{\bar{x}} \in V^* \Leftrightarrow$ $\begin{cases} p \text{ variables de base ont la valeur 1} \\ 1 \text{ variable de base est nulle (forme 1)} \end{cases}$

Le problème (B) est ainsi résolu.

2. - $\bar{\bar{x}} \notin V^*$ <=> [p - 1 variables de base ont la valeur 1
2 variables de base ont leurs valeurs comprises entre 0 et 1 (forme 2)

c) Il est très simple de déterminer une solution de base réalisable avec une matrice de base facilement inversible, du type classique :

α	β	
U'	γ	0
0		U

avec [U, U' matrices unité, α vecteur ligne
γ vecteur colonne, $\beta \in \mathbb{R}$

Mais, il faut bien noter que, par construction de la contrainte (54), toute solution réalisable de $(\overline{PB})$ est solution réalisable de $(\bar{B})$ alors que la réciproque est fausse. (le système (46) est remplacé par la seule contrainte (54) (combinaison linéaire des contraintes initiales) sans modification du nombre des variables). D'où la relation :

$$v(\bar{B}) \geq v(\overline{PB}) \geq v(B)$$

qui est à la base de notre décision de choisir $(\overline{PB})$; $v(\overline{PB})$ sera, à l'évidence, obtenue beaucoup moins rapidement que $v(\bar{B})$($(\overline{PB})$ possède plus de contraintes que $(\bar{B})$)mais correspondra à une meilleure évaluation de $v(B)$.

Pour les problèmes traités sur ordinateur, on verra au § II. 3 que, du point de vue temps de calcul, la perte initiale découlant du choix de $(\overline{PB})$ est largement compensée par le gain obtenu lors de l'exploration des solutions de (B). Pour ces problèmes on aura toujours $v(\bar{B}) >> v(\overline{PB})$,mais ce résultat n'est pas général : voir les exemples du § I.4 du chapitre 2.

II.2.2.2 - REDUCTION DE LA TAILLE DE (B)

Etant donnés

(i) $\underline{v}(B) = c\underline{x}$ où $\underline{x} \in F(B)$ est obtenue par l'intermédiaire de la résolution du système $\begin{cases} a_{J_1}\, y = b \\ \sum_{j \in J_1} y_j = p \\ 0 \le y_j \le \min(p, \lfloor b/a_j \rfloor)\ \forall j \in J_1 \end{cases}$

(ii) $I = \{1,\ldots,p+1\}$ (resp. $\bar{I} = \{1,\ldots,n\} \backslash I$) l'ensemble d'indices de base (resp. hors base) et le vecteur ligne d des coefficients égaux aux coûts réduits fournis par la méthode simpliciale à l'optimum de $(\overline{PB})$

le corollaire du théorème 5 (chapitre 1) permet de conclure que :

Théorème 31

Si $\exists j \in \bar{I} : \lfloor v(\overline{PB}) + d_j \rfloor \le \underline{v}(B)$ *alors* $x_j \to 0$

Note

Par la suite n désignera toujours le nombre de variables même s'il y a eu élimination de certaines d'entre elles.

□

II.2.2.3 - EXPLORATION DE V^*

a) Schéma d'exploration - choix d'une partition de $\{1,\ldots,n\}$:

Les colonnes des contraintes sont classées suivant l'ordre décroissant des valeurs absolues des coûts réduits. Si J_k représente l'ensem

J_k modifié par ce classement (k = 1,...,p), le problème (B) se formule finalement :

$$\left[\begin{array}{l} \max \sum_{j=1}^{n} c_j x_j \\ \text{s.c.} \sum_{j=1}^{n} a_j x_j = b \\ \sum_{j \in J_k} x_j = 1 \quad k = 1,\ldots,p \\ x \in V \end{array}\right. \qquad (57)$$

avec $|d_n| \geq |d_{n-1}| \geq \ldots \geq |d_{r+1}| \geq d_r = d_{r-1} = \ldots = d_1 = 0$ où $r = p + m$

Note

Même dans le cas où certaines composantes $d_j (j \in \bar{I})$ sont nulles, on conviendra que les indices 1, 2,...r sont les indices de base.

□

Le principe d'exploration systématique de notre méthode est le suivant :

(i) à partir de x_I^1 défini par :

$$x_j^1 = 0 \quad j = 1,\ldots,n' = n - r - 1$$

les $2^{n'}-1$ autres sommets x_I^j $(j = 2, 3,\ldots,2^{n'})$ du cube unité de $\mathbb{R}^{n'}$ sont générés dans l'ordre lexicographique :

sommets \ Composantes	n'	n'-1	.	.	3	2	1
2	0	0	.	.	0	0	1
3	0	0	.	.	0	1	0
4	0	0	.	.	0	1	1
5	0	0	.	.	1	0	0
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
$2^{n'}$	1	1	.	.	1	1	1

pour ne considérer que les éléments vérifiant les relations (57).

Par définition de V^*, il est clair que

Propriété 8

Une condition nécessaire pour que $x \in V^*$ *est que* x *possède* p *composantes égales à 1.*

La condition est suffisante lorsque ces p *composantes* $x_{j_1},\ldots,x_{j_p}$ *sont telles que* $j_k \in J_k$ $(k = 1,\ldots,p)$

Note

La détermination de la première solution réalisable de (B) dans l'ordre lexicographique est accélérée par les déterminations de $t^* = \max \{t_1, t_2\}$:

avec $t_1 = \max_{1 \le k \le p} \min \{j \mid j \in J_k\}$

et $t_2 = \min \{t \mid Z(t) = b\}$

où $Z(t) = \max \{ \sum_{j=1}^{t} a_j x_j \mid \sum_{j=1}^{t} a_j x_j \leq b \; ; \; \sum_{j=1}^{t} x_j \leq p \; ; \; x_j = 0 \text{ ou } 1 \; j=1,\ldots,t\}$.

□

(ii) A chaque étape de ce schéma exploratoire si $\underline{x}_j (j = r+2,\ldots,n)$ désignent les valeurs respectives des variables d'indices r+2, r+3,...,n, il faut résoudre un problème du type :

$$(PA) \quad \left[\begin{array}{l} \max \sum_{j=1}^{r+1} c_j x_j \\ \text{s.c.} \sum_{j=1}^{r+1} a_j x_j = b - \sum_{j=r+2}^{n} a_j \underline{x}_j \\ \sum_{j \in I_k} x_j = \delta_k \quad k = 1,\ldots,p \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,r+1 \end{array}\right.$$

avec

$$\left[\begin{array}{l} I_k = \tilde{J}_k \cap (I \cup \{r+1\}) \\ \delta_k = \begin{cases} 0 \text{ si } \exists j \in \tilde{J}_k - I_k \text{ tel que } \underline{x}_j = 1 \\ 1 \text{ sinon} \end{cases} \end{array}\right.$$

Note

Il est à noter que $I \cap J_k \neq \emptyset$ $k = 1,\ldots,p$. Ainsi la sous-matrice composée des p dernières lignes de la matrice des contraintes de (PA), comporte un seul élément non nul (égal à 1) dans chaque colonne et au moins un élément non nul (égal à 1) dans chaque ligne.

□

Théorème 32

S'il existe une solution $\underline{x} \in F(B)$: $c\underline{x} = \underline{v}(B)$ *telle que*

$$\exists k \in \{r+1,\ldots,n\} \quad \begin{cases} \lfloor v(\overline{PB}) + d_k \rfloor \leq \underline{v}(B) \\ x_k = 1 \end{cases}$$

alors cette solution est une solution optimale de (B) $(v(B) = \underline{v}(B))$.

Démonstration : Voir théorème 2 de [G13].

□

b) Exploration implicite des éléments de V^*:

Tant que les hypothèses du théorème 32 ne sont pas vérifiées, il faut déterminer des critères permettant d'éliminer des familles de solutions. Pour exposer ces tests, on suppose être à l'étape suivante de l'algorithme :

> Etant donnés X_0 et X_1 : $X_0 \cup X_1 = \{t, t+1,\ldots,n\}$ ($\Rightarrow X_2 = \{1,\ldots,t-1\}$) toutes les solutions telles que $x_j = 1$ $\forall j \in X_1$ et $x_j = 0$ $\forall j \in X_0$ ont été considérées.

Compte-tenu de l'ordre lexicographique, il faut envisager la modification de la variable x_t ; deux cas peuvent se produire :

> cas 1 : $t \in X_1$: il faut considérer la variable d'indice $t+1$ (X_1 devient $X_1 \setminus \{t\}$)
>
> cas 2 : $t \in X_0$: si $\mathcal{J}_k$ désigne l'ensemble auquel appartient t, deux possibilités sont à étudier :
>
> (i) $\exists j \in \mathcal{J}_k \cap X_1$: il faut considérer la variable d'indice $t+1$ (X_0 devient $X_0 \setminus \{t\}$).
>
> (ii) $\mathcal{J}_k \cap X_1 = \emptyset$: pour explorer systématiquement toutes les solutions de V^*, il faut, a priori, attribuer la valeur 1 à la variable d'indice t (X_1 devient $X_1 \cup \{t\}$ et X_0 devient $X_0 \setminus \{t\}$) pour envisager la résolution du sous-problème :

$$(P)\quad \left[\begin{array}{l} \max \sum_{j=1}^{t-1} c_j x_j \\ \text{s.c.} \sum_{j=1}^{t-1} a_j x_j = b(P) \qquad (58) \\ \sum_{j\in I_k(t)} x_j = \delta_k(t)\ k = 1,\ldots,p \qquad (59) \\ x_j = 0 \text{ ou } 1\ j = 1,\ldots,t-1 \end{array}\right.$$

avec

$$\left[\begin{array}{l} b(P) = b - \sum_{j\in X_1} a_j \\ I_k(t) = \{1, 2,\ldots,t-1\} \cap \mathcal{J}_k \\ \delta_k(t) = \begin{cases} 0 \text{ si } \exists j \in \mathcal{J}_k - I_k(t) \text{ tel que } x_j = 1 \\ 1 \text{ sinon} \end{cases} \end{array}\right.$$

Deux types de tests sont considérés :

1) Tests basés sur les contraintes :

* Il est clair que la contrainte (58) ne peut admettre de solution bivalente si l'une ou l'autre des hypothèses suivantes est vérifiée :

H1 : le P.G.C.D. des a_j ($j = 1,\ldots,t-1$) ne divise pas $b(P)$

H2 : en posant

$$\left[\begin{array}{l} \alpha = a_s = \min_{1\le j\le t-1} a_j \\ \beta = \min_{\substack{1\le j\le t-1 \\ j\neq s}} a_j \end{array}\right.$$

on a à la fois

$b(P) < \beta$, $b(P) \neq 0$ et $b(P) \neq \alpha$

* Posons $N = p - |X_1|$. Les contraintes (59) impliquent qu'une solution réalisable x de (P) doit être telle que $|\{j \in \{1,\ldots,t-1\} | x_j = 1\}| \leq N$

On considère alors le problème

$$\begin{cases} z = \max \sum_{j=1}^{t-1} a_j x_j \\ \text{s.c.} \sum_{j=1}^{t-1} a_j x_j \leq b(P) \\ |\{j \in \{1,\ldots,t-1\} | x_j = 1\}| \leq N \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,t-1 \end{cases}$$

Si $z < b(P)$ alors on peut conclure que $F(P) = \emptyset$

2) Test basé sur la fonction économique

Le majorant de v(P) choisi est $v(RPB(\bar{w}) \mid x_j = 0 \ \forall j \in X_0, x_j = 1 \ \forall j \in X_1)$ où $\bar{w}$ désigne le multiplicateur associé à $\tilde{L}x = u$ à l'optimum de $(\overline{PB})$.

II.3 - EXPÉRIENCES NUMÉRIQUES

La méthode, programmée sur GAMMA M 40, environ 65 fois plus lent que l'UNIVAC 1110, a été testée sur des problèmes de 80, 112 et 160 variables.

Pour chacun de ces problèmes le nombre de périodes est identique ($m = 4$). Seuls le nombre p d'emplacements et le nombre q de moyens à installer changent d'un problème à l'autre.

La matrice d'affectation représentant les 16 chroniques d'installation envisageables sur un emplacement est donc :

$A_1 = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1)$
$A_2 = (0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1)$
$A_3 = (0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1)$
$A_4 = (0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1)$

Les coefficients $a_j (j = 1,\ldots,16)$ et b obtenus par contraction sont répertoriés pour chacun des problèmes tests dans le tableau 42 (un exemple numérique relatif au problème de 80 variables est fourni en note finale).

Dans le tableau 43 sont rassemblés pour chacun des problèmes (B), $(\bar{B})$ et $(\overline{PB})$ les valeurs optimales de la fonction économique (colonne v.o.), les temps de calcul donnés en secondes (colonnes t, t1, t2) ainsi que les nombres de solutions réalisables testées lors de l'exploration de (B) (colonnes SR1 et SR2).

LILLE I

Les colonnes SR1 et t1 (resp. SR2 et t2) sont relatives à la résolution de (B) en ayant résolu dans une première phase le problème $(\overline{PB})$ (resp. $(\bar{B})$). Chaque temps de calcul indiqué pour (B) est la somme du temps de résolution en continu et du temps d'exploration des solutions bivalentes.

La résolution de $(\bar{B})$ en première phase ne permet pas d'obtenir l'optimum de (B) dans les limites de temps de calcul indiqués dans le tableau ; dans les cases correspondantes de la colonne SR2 sont alors indiqués en partie supérieure le nombre de solutions réalisables testées et en partie inférieure la valeur de la fonction économique atteinte dans ces limites.

Problèmes		a_j $j = 1,\ldots,16$																b
1	80 v P=5 a_i=2	0	64	16	80	4	68	20	84	1	65	17	81	5	69	21	85	170
2	112 v P=7 a_i=i+1	0	141	30	171	6	147	36	177	1	142	31	172	7	148	37	178	845
3	160 v P=10 a_i=4	0	343	49	392	7	350	56	399	1	344	50	393	8	351	57	400	1600

Tableau 42

Problèmes	$(\overline{PB})$		(B)					$(\bar{B})$	
	v.o.	t	SR1	t1	v.o.	t2	SR2	t	
1	4 844.66	20	1	21	4 847	179	5	4	1
2	8 273 Sol.bivalente	25		25	**8 273**	> 2 h	8 8 346	8.7	3
3	9 739.33	55	1	59	9 740	> 2 h	4 10 023	17.3	2

Tableau 43

Note

Le système de contraintes du problème 1 ($p = 5$; $a_i = 2$ $i = 1,2,3,4,\ldots$) est de la forme

$$(S_1) \qquad \left[\begin{array}{ll} L_i \sum_{k=1}^{5} x_{J_k} = 2 & i = 1,\ldots,4 \\ \sum_{j \in J_k} x_j = 1 & k = 1,\ldots,5 \\ x_j = 0 \text{ ou } 1 & j = 1,\ldots,80 \end{array}\right] \equiv x \in V^*$$

En posant

$$\phi_i(x) = 2 - L_i \sum_{k=1}^{5} x_{J_k} \qquad i = 1,\ldots,4$$

et en remarquant que

$$\forall i \quad \begin{array}{l} \max\{\phi_i(x) \mid x \in V^*\} = 2 \\ \min\{\phi_i(x) \mid x \in V^*\} = -3 \end{array}$$

la contraction de tout système du type

$$\left[\begin{array}{l} \psi(x) = 0 \\ \phi_i(x) = 0 \\ x \in V^* \end{array}\right.$$

sera d'après le théorème 6 (chapitre 1),

$$\psi(x) + 4\,\phi_i(x) = 0$$

Le système équivalent à (S_1) est donc le suivant

$$\begin{cases} (L_1 + 4L_2 + 16L_3 + 64L_4) \cdot \sum_{k=1}^{5} x_{J_k} = 170 \\ x \in V^* \end{cases}$$

La forme particulière des contraintes fait que les résultats dûs à Bradley ne fournissent pas des coefficients de taille plus faible (voir [G13]).

□

III

RESOLUTION D'UN

PROBLEME DE FABRICATION

III.1 - PROBLÈME ÉTUDIÉ

III.1.1 - Enoncé

Le problème consiste à couler des "brames" (tôles préliminaires de forte épaisseur) afin d'obtenir par transformation (laminage puis découpage) tout ou partie d'une commande c'est-à-dire un ensemble de t tôles de dimension et de qualité exigées.

Une tôle k de caractéristique (i', j') où i' désigne les dimensions et j' la qualité du matériau, peut être réalisée à partir de coulées de dimensions et de qualités supérieures. Donc le coût de fabrication de la tôle k est fonction des caractéristiques de réalisations (i,j) [nécessairement supérieures à (i', j')].

De plus à un ensemble de brames de caractéristiques (i,j) produit par une séquence de coulées identiques et en nombre u_{ij}, correspondront $(u_{ij}-1)$ coulées pleines (de poids P), seule la dernière coulée pouvant être partiellement utilisée (de poids compris entre P et $P-M_{ij}$, la marge M_{ij} étant imposée).

Ainsi connaissant les coûts et contraintes matérielles d'utilisation des procédés de fonderie employés, il faut réaliser le meilleur profit c'est-à-dire une pondération entre le maximum de tôles fabriquées et le minimum de rebus résultant.

III.1.2 - Formulation

A chaque pièce k on associe une variable d'état bivalente.

$$x_{ijk} = \begin{cases} 1 \text{ si la pièce commandée k est fabriquée dans la dimension i et la qualité j, le poids de matériau nécessaire étant noté } a_{ijk} \\ 0 \text{ sinon} \end{cases}$$

Le fait qu'il y ait possibilité de rejeter les commandes non rentables et que le choix d'une dimension et d'une qualité implique l'élimination des autres possibilités, se résume par la contrainte

$$\sum_{i,j} x_{ijk} \leq 1 \quad \forall k$$

En posant :

p : le nombre de pièces commandées

s : nombre total de dimensions

I(k) : l'ensemble des dimensions $i \in \{1,\ldots,s\}$ acceptables pour la pièce k

$K(i,j) = \{k \in \{1,\ldots,p\} \mid a_{ijk} \neq 0\} \quad \forall (i,j)$

J(k) : l'ensemble des qualités j acceptables pour la pièce k.

$$(PF) \quad \left[\begin{array}{l} \text{maximiser} \sum_{i,j,k} c_{ijk}\, x_{ijk} - \alpha \sum_{i=1}^{s} \delta_i \\ \text{sous les conditions} \\ \sum_{(i,j) \in I(k) \times J(k)} x_{ijk} \leq 1 \quad k=1,\ldots,p \qquad (60) \\ Pu_{ij} - M_{ij} \leq \sum_{k \in K(i,j)} a_{ijk}\, x_{ijk} \leq Pu_{ij} \quad \forall i \forall j \qquad (61) \\ \delta_i = \begin{cases} 1 \text{ si } \exists (i,j) : u_{ij} \neq 0 \\ 0 \text{ sinon} \end{cases} \qquad (62) \\ u_{ij} \in \mathbb{N} \quad x_{ijk} = 0 \text{ ou } 1 \quad \forall i \forall j \forall k \end{array}\right.$$

avec les données

$$\alpha, P, M_{ij}, c_{ijk}, a_{ijk} \in \mathbb{R} \quad \forall i \forall j \forall k$$

Notes

(i) $\delta_i = \begin{cases} 1 \text{ si au moins une coulée est réalisée dans la dimension i} \\ 0 \text{ sinon} \end{cases}$

(ii) Des conditions de bornes sur les u_{ij} peuvent éventuellement être introduites.

(iii) Les dimensions de (PF) sont

- Pour les variables

$$n = \sum_{k=1}^{P} |J_k| + \sum_{i=1}^{s} |J(i)| + s$$

où $J_k = \{(i,j) \mid i \in I(k),\ j \in J(k)\}$ (i.e. ensemble des brames (i,j) acceptables pour la commande k)

J(i) : ensemble des qualités j compatibles avec la dimension i.

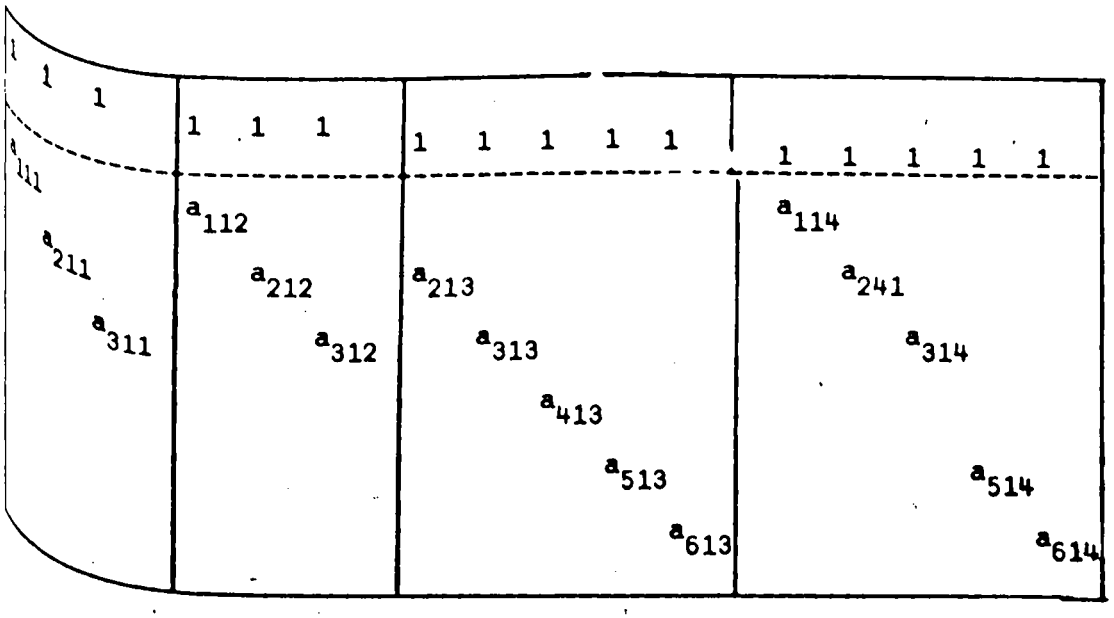
- Pour les contraintes : p + m

avec $m = \sum_{i=1}^{s} |J(i)|$

(iv) A désignera la matrice des contraintes (61) qui est composée de sous-matrices "pseudo-diagonales"

[matrices diagonales agrandies de lignes nulles c'est-à-dire
- qu'il existe un élément non nul et un seul par colonne
- que les éléments non nuls $A_{i_1}^{j}$ et $A_{i_2}^{j+1}$ de deux colonnes consécutives sont tels que $i_1 < i_2^1$].

A titre d'exemple, un petit problème, relatif à 4 pièces, 6 dimensions et une qualité correspond à la structure matricielle suivante (pour les contraintes (60) et (61)).



(60)

(61)

III.2 - Résolution de (PF)

Les différentes particularités de la méthode sont les suivantes.

III.2.1 - Réduction "triviale" de la taille de (PF)

Une coulée de caractéristiques (i,j) ne peut être réalisée si

$$\sum_{k\in K(i,j)} a_{ijk} < P - M_{ij}$$

ce qui implique *l'élimination de la contrainte* correspondante du système (61) et la *fixation définitive* à la valeur 0 des *variables* associées c'est-à-dire

$$x_{ijk} = 0 \quad \forall k \in K(i,j)$$

III.2.2 - Minorant de v(PF)

Un minorant de la valeur de (PF) est déterminé par des résolutions successives de problèmes du knapsack. L'algorithme (à rapprocher de [G42]) est le suivant :

En notant

$$(B_{ij}) \begin{cases} \max \sum_{k\in K(i,j)} c_{ijk}\, x_{ijk} \\ \text{s.c.} \sum_{k\in K(i,j)} a_{ijk}\, x_{ijk} + y_{ij} \equiv 0 \pmod P \\ 0 \le y_{ij} \le M_{ij},\ y_{ij} \in \mathbb{N} \\ x_{ijk} = 0 \text{ ou } 1 \quad \forall k \in K(i,j) \end{cases}$$

et x^{ij} une solution optimale associée, s algorithmes gloutons G_i (un par dimension i) de valeur $v(G_i)$ sont mis en jeu, pour poser finalement

$$\underline{v}(PF) = \max_{i=1,\ldots,s} v(G_i)$$

Algorithme G_i

0 $Z \leftarrow 0$; $K \equiv \emptyset$; $S \equiv \{1,\ldots,s\}\backslash\{i\}$
$(P_{ij}) \equiv (B_{ij} \mid x_{ijk} = 0 \quad \forall k \in K)$

1 si $\forall j \in J(i)$ $F(P_{ij}) = \emptyset$ alors aller en 3

2 soit j^* : $v(P_{ij}^*) = \max_{j \in J(i)} v(P_{ij})$

2.1 $Z \leftarrow Z + v(P_{ij}^*)$
$K \equiv K \cup \{k \mid x_{ij^*k} = 1\}$
aller en 1

3 si $\forall i \in S$ $\forall j \in J(i)$ $F(P_{ij}) = \emptyset$ alors aller en 4
soit (i^*, j^*) : $v(P_{i^*j^*}^*) = \max_{i \in S} \max_{j \in J(i)} v(P_{ij})$

$S \equiv S\backslash\{i^*\}$
$i \leftarrow i^*$
aller en 2.1

4 $v(G_i) \leftarrow Z + \alpha(s - |S|)$; stop

Note

En posant $h = \sum_{k \in K(i,j)} a_{ijk}/P$

si $\sum_{k \in K(i,j)} a_{ijk} - \lfloor h \rfloor P \geq P - M_{ij}$ alors $v(B_{ij}) = \sum_{k \in K(i,j)} c_{ijk}$

sinon le problème (B_{ij}) est équivalent au problème du knapsack classique

$$\left[\begin{array}{l} \max \sum_{k \in K(i,j)} c_{ijk}\, x_{ijk} \\ \text{s.c.} \sum_{k \in K(i,j)} a_{ijk}\, x_{ijk} \leq \lfloor h \rfloor P \\ x_{ijk} = 0 \text{ ou } 1 \quad \forall k \in K(i,j). \end{array}\right.$$

□

III.2.3 - Résolution de $(\overline{PF})$

Pour résoudre $(\overline{PF})$, il suffit de remplacer les conditions $x_{ijk} = 0$ ou 1 par $x_{ijk} \leq 1$ et les conditions (62) par

$$\begin{cases} \delta_i \leq 1 \\ b_i \, \delta_i \leq \sum_{j \in J(i)} u_{ij} \leq B_i \, \delta_i \end{cases} \qquad i = 1,\ldots,s$$

avec

$\forall i$ b_i = réel suffisamment petit (1/1000 par exemple)

B_i = nombre maximum de coulées réalisables dans la dimension i c'est-à-dire, en posant $Q_i = \sum_{k=1}^{P} \max_{j \in J(i)} a_{ijk}$ $\forall i$,

$$B_i = \begin{cases} Q_i/P & \text{si } Q_i - \lfloor Q_i/P \rfloor \geq P - \max_{j \in J(i)} M_{ij} \\ \lfloor Q_i/P \rfloor & \text{sinon} \end{cases}$$

Note

En posant $Q = \sum_{k=1}^{P} \max_{i \in I(k)} \max_{j \in J(i)} a_{ijk}$, pour déterminer

$$B = \begin{cases} Q/P & \text{si } Q - \lfloor Q/P \rfloor \geq P - \max_{i=1,\ldots,s} \max_{j \in J(i)} M_{ij} \\ \lfloor Q/P \rfloor & \text{sinon} \end{cases}$$

(nombre maximum de coulées réalisables toutes dimensions confondues), [13] contrainte suivante

$$\sum_{i,j} u_{ij} \leq B$$

pourrait être adjointe à celle de $(\overline{PF})$.

□

III.2.4 - Réduction de la taille de (PF)

L'originalité de la structure de (PF) fait que l'application du théorème 31 (partie II) pour les variables δ_i permet l'élimination en bloc de contraintes et de variables.

En appelant z_i la variable d'écart associée à δ_i dans la contrainte $\delta_i \leq 1$, en notant d_{δ_i} et d_{z_i} les coûts réduits associés à l'optimum de $(\overline{PF})$ et en posant

$$\gamma_i = \begin{cases} d_{z_i} & \text{si } \delta_i \text{ est de base à l'optimum de } (\overline{PF}) \\ d_{\delta_i} & \text{sinon} \end{cases}$$

Si $\exists\, i_o$: $\lfloor v(\overline{PF}) - |\gamma_{i_o}| \rfloor \leq \underline{v}(PF)$ alors

- si δ_{i_o} est de base, au moins l'une des contraintes (60) concernées par la dimension doit être active (i.e. au moins une pièce doit être fabriquée dans la dimension i_o).

- si δ_{i_o} est hors base, toutes les variables et contraintes concernées par la dimension i_o doivent être éliminées (i.e. aucune pièce ne doit être fabriquée dans la dimension i_o).

III.2.5 - Schéma exploratoire

a) la partition $\{I, \bar{I}\}$ de $\{1,\ldots,n\}$ est définie par

- $|I| = m$

- A^I diagonale régulière

ce qui permet à chaque étape la résolution d'un système simple du type

$$\begin{cases} A^{I}_{ij}\, x_I + y_{ij} \equiv - A^{\bar{I}}_{ij}\, x_{\bar{I}} \qquad (\text{mod } P) \\ x_\ell = 0 \text{ ou } 1 \quad \forall \ell \in I \\ 0 \le y_{ij} \le M_{ij} \end{cases}$$

b) les variables sont classées "dimension qualité" par "dimension qualité", dans le but de traiter globalement les variables relatives à une même dimension. Plus précisément, ce classement est de telle sorte que la "dimension qualité" dans laquelle il est possible d'effectuer la plus grande séquence de coulées soit considérée en priorité ; et ainsi de suite, par ordre décroissant d'intérêt jusqu'à la "dimension qualité" correspondant à la plus petite séquence de coulées.

III.3 - Expériences numériques

Les caractéristiques des huit problèmes testés sur UNIVAC 1110 sont fournies au tableau 44 ; pour chacun d'eux, la première (resp. seconde) ligne concerne les caractéristiques initiales (resp. après réduction "triviale" de leur taille [voir § III.2.1]) ; les nombres de contraintes et de variables sont ceux de $(\overline{PF})$; les vecteurs de la colonne "Nombre de contraintes (61)" indiquent pour chaque dimension i : $\left|\{j \in J(i) \mid \sum_k a_{ijk} \ge P - M_{ij}\}\right|$.

Les temps de calcul donnés en secondes répertoriés dans le tableau 45 ; il est à noter que l'exploration a été interrompue pour les problèmes 7 et 8 après un temps d'exécution de 300 secondes, alors que de bons minorants de v(PF) sont obtenus par l'algorithme du § III.2.2 en moins de 40 secondes.

Nombre p pièces	Nombre s de dimensions	Nombre de coulées par dimension	Nombre Total de coulées	Nombre de contraintes (61)	(60)+(61)	$\overline{(PF)}$	Nombre de variables x_{ijk}	total
4	6	(0,0,1,0,0,1)	1	24 (4,4,4,4,4,4)	28	64	50	80
	2			2 (0,0,1,0,0,1)	6	12	8	12
6	6	(1,1,1,0,0,0)	1	6 (1,1,1,1,1,1)	12	30	18	31
	3			3 (1,1,1,0,0,0)	9	18	14	20
11	6	(1,1,1,1,0,0)	1	72 (12,12,12,12,12,12)	90	174	960	1038
	4			48 (12,12,12,12, 0, 0)	66	78	792	844
21	6	(3,2,2,1,0,2)	3	57 (10,10,10,10,7,10)	78	147	282	345
	5			24 (6,6,6,2,0,4)	43	58	205	234
24	6	(2,1,1,1,0,0)	2	24 (4,4,4,4,4,4)	48	84	328	358
	4			16 (4,4,4,4,0,0)	40	52	276	296
29	5	(2,3,2,3,0)	3	20 (4,4,4,4,4)	48	78	373	398
	4			16 (4,4,4,4,0)	44	56	349	369
45	6	(4,4,3,3,2,1)	6	60 (10,10,10,10,10,10)	105	177	610	676
	6			42 (8, 9, 9, 9, 6, 1)	87	105	552	600
52	6	(2,1,4,4,4,2)	6	44 (4,8,8,8,8,8)	96	152	701	751
	6			32 (4,4,8,8,4,4)	84	102	625	663

Tableau 44

Pb n°	Réduction "Triviale" et partitionnement	Relaxation	Réduction et Exploration	Total
1	0.64	0.12	0.09	0.85
2	0.60	0.19	0.09	0.88
3	5	145	12	162
4	2.70	16.70	4	23.40
5	2.80	15.70	30	48.50
6	2.70	28	40	70.70
7	7	173	> 300	> 300
8	5	157	> 300	> 300

Tableau 45

UN TYPE DE PROBLEME

DU

KNAPSACK NON LINEAIRE

CHAPITRE 4

I - INTRODUCTION

Ce chapitre concerne l'étude d'une extension du problème du knapsack en variables bivalentes classique , que nous appellerons "knapsack baudruche" (en anglais "collapsing knapsack"), introduit par Guignard et Rosner dans [EK9].

Le second membre de la contrainte de ce knapsack "baudruche" n'est pas constant : c'est une fonction non croissante du nombre de variables mises à un (i.e. du nombre d'objets mis dans le sac) ; la formulation de ce problème est donc la suivante (lorsque le nombre de variables est n) :

$$(K) \quad \left[\begin{array}{l} \max \ \sum_{j=1}^{n} c_j x_j \\ \text{s.c.} \ \sum_{j=1}^{n} a_j x_j \leq b\left(\sum_{j=1}^{n} x_j\right) \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,n \end{array}\right.$$

avec $\left\{\begin{array}{l} c_j, a_j \in \mathbf{N} \quad \forall\, j \in \{1,\ldots,n\} \\ b : \text{application non croissante de } \{1,\ldots,n\} \text{ dans } \mathbf{N}. \end{array}\right.$

Une interprétation concrète de (K) est la généralisation de celle du knapsack classique lorsque le conditionnement des objets mis dans le sac, est pris en compte.

Un autre aspect de l'intérêt de ce problème concerne l'exploitation optimale d'un ordinateur en multiprogrammation :

On suppose qu'à un instant donné, un choix étant à faire entre n programmes d'utilisateurs potentiels P_j, $j = 1,\ldots,n$, on connaisse :

a_j : ressources demandées par P_j (i.e. programme objet, y compris les sous-programmes de bibliothèque)

c_j : profit réalisé par l'exploitant pour le passage de P_j

R : ensemble des ressources disponibles par l'ordinateur.

Le passage simultané de k programmes $P_1, P_2, \ldots, P_k$ nécessite en fait un encombrement $\sum_{j=1}^{k} a_j + s(k)$ où s(k) désigne celui nécessaire au système pour la gestion du matériel et l'exécution des k programmes. En représentant ce "besoin système" par une fonction non décroissante du nombre k d'utilisateurs $\bar{s}(k)$ - obtenue statistiquement - de telle sorte que l'ensemble des ressources disponibles lors du passage de k programmes est $b(k) = R - \bar{s}(k)$, on retrouve la formulation de (K) avec x_j variable d'état prenant la valeur 1 lorsque P_j est "actif", et 0 sinon ($j = 1, \ldots, n$).

La méthode de réduction de la taille de (K) présentée en partie IV, inclue

(i) la détermination d'un encadrement de la somme des variables (voir § III.1 pour l'aspect théorique et § IV.2 pour l'exposé des quatre algorithmes associés qui sont tous de complexité O(n)).

(ii) la construction d'une enveloppe convexe de points de $\mathbb{Z} \times \mathbb{Z}$ (voir § IV.3) qui est à la base de la détermination de deux types de problèmes du knapsack classiques (§ III.2)

- le premier (appelé linéarisation minorante) permet la génération d'un algorithme glouton pour une borne inférieure de la valeur de (K) (§ IV.4).

- la résolution de la relaxation - complexité O(n) - du second (appelé linéarisation majorante) fournit une borne supérieure de la valeur de (K) et conduit à un schéma de réduction utilisant la relaxation lagrangienne et la contraction de contraintes (§ IV.5).

Les résultats numériques de la partie V montrent que la méthode proposée donne des résultats satisfaisants quant aux nombres de variables éliminées et à l'évolution des temps de calcul avec la taille des problèmes :

- elle suffit à elle seule dans de nombreux cas - et en particulier pour les cinq problèmes tests de [EK9] - pour fournir soit la solution optimale, soit un sous-problème dont la solution sera obtenue par une technique simple d'énumération.

- l'apport de cette réduction en phase initiale de la méthode proposée dans [EK9] permet une accélération importante de la résolution : les temps d'exécution sont couramment divisés par 100 et plus.

II - NOTATIONS

Les notations suivantes viennent en complément de celles décrites en tout début d'exposé :

p_j : point de $\mathbb{R}^2$ de coordonnées $(j,b(j))$ $j \in \{1,\ldots,m\}$ où $m = \max \{s \in \{1,2,\ldots,n\} \mid b(s) \neq 0\}$

$M_{\ell,u} = \{p_\ell, p_{\ell+1},\ldots,p_u\}$ où $1 \leq \ell < u \leq m < n$

$H_{\ell,u}$: Profil du convexe $[M_{\ell,u}]$

$(p_j\ p_k)$: droite déterminée par p_j et p_k

$[p_j\ p_k]$: segment de support $(p_j\ p_k)$

III - RESULTATS PRELIMINAIRES

Les résultats théoriques nécessaires à l'exposé des algorithmes d'encadrement de $\sum_j x_j$ (§ III-1) précèdent la présentation des linéarisations du problème qui seront utilisées non seulement pour obtenir des bornes inférieure et supérieure de v(K) mais également pour réduire la taille de (K).

III.1 - ENCADREMENT DE $\sum_{j=1}^{n} x_j$

Les quatre méthodes d'encadrement de $\sum_{j=1}^{n} x_j$ - i.e. de détermination de $\ell, u \in \mathbb{N}$: $\ell \leq \sum_{j=1}^{n} x_j \leq u$ $\forall$ x réalisable et/ou optimale de (K) - présentées en § IV.2 sont justifiées par les résultats des cinq théorèmes suivants :

Théorème 33

Si $\exists\, U \subset \{1,\ldots,n\}$:

$$\begin{cases} \forall\, j \in U \quad c_j \geq c_k \quad \forall\, k \notin U & (63) \\ \sum_{j \in U} a_j \leq b(|U|) & (64) \end{cases}$$

alors $\forall\, x \in V$: $cx > \sum_{j \in U} c_j \Longrightarrow \sum_{j=1}^{n} x_j \geq |U| + 1.$

Démonstration :

Etant donné U vérifiant (64), la solution $\underline{x}$ définie par $\underline{x}_j = 1$ $\forall\, j \in U$, $\forall\, j \notin U$, est réalisable pour (K) puisque $a\underline{x} \leq b(|U|)$ par hypothèse et $\sum_{j=1}^{n} \underline{x}_j = |U|$ par construction $\Longrightarrow$ $c\underline{x} = \sum_{j \in U} c_j$ est une borne inférieure de v(K).

Si de plus U vérifie (63), alors

$$\forall\, x \in V \quad : \sum_{j=1}^{n} x_j \leq |U| \Longrightarrow cx \leq c\underline{x}$$

d'où la conclusion. □

Note : L'algorithme B_1 associé cherchera l'ensemble U (vérifiant (63) et (64)) de cardinalité maximale pour poser $\ell = |U| + 1$. □

Le résultat suivant est une généralisation de celui de [G25] rappelé dans [K11]

Théorème 34

Etant donné $\underline{v}(K)$

$$\textit{si } \exists\, U \subset \{1,\ldots,n\} : \begin{cases} \forall\, j \in U \quad c_j \geq c_k \quad \forall\, k \notin U & (65) \\ \sum_{j\in U} c_j \geq \underline{v}(K)+1 > \sum_{j\in U} c_j - \min_{j\in U} c_j & (66) \end{cases}$$

$$\textit{alors } \forall\, x \in V \quad : cx > \underline{v}(K) \Longrightarrow \sum_{j=1}^{n} x_j \geq |U|.$$

Démonstration :

Etant donné U vérifiant (66), la solution $\underline{x}$ définie par $\underline{x}_j = 1\ \forall\, j \in U$, $0\ \forall\, j \notin U$, permet d'améliorer la meilleure valeur connue de la fonction économique $\underline{v}(K)$ supposée connue, puisque $c\underline{x} \geq \underline{v}(K) + 1$.

Si de plus U vérifie (65), alors

$$\forall\, x \in V \quad : \sum_{j=1}^{n} x_j < |U| \Longrightarrow cx \leq \underbrace{c\underline{x} - \min_{j\in U} c_j}_{(66)} \Longrightarrow cx \leq \underline{v}(K)$$

d'où la conclusion. □

Note : L'algorithme B_2 associé cherchera donc les $|U|$ plus grands éléments de la suite (c_j) - avec U vérifiant (66)- pour poser $\ell = |U|$. □

Le théorème suivant généralise un résultat de [G25] au cas de la contrainte du knapsack "baudruche" ; il est également utilisé dans [EK9] (avec toutefois le classement par ordre croissant des éléments de la suite (a_j)) :

Théorème 35

$$\text{Si } \exists\, L \in \{1,\dots n\} : \begin{cases} \forall\, j \in L \;\; a_j \le a_k \quad \forall\, k \notin L & (67) \\ \sum_{j\in L} a_j \le b(|L|) & (68a) \\ b(|L|+1) < \sum_{j\in L} a_j + \min_{j\notin L} a_j & (68b) \end{cases}$$

$$\text{alors } \forall\, x \in F(K) \sum_{j=1}^{n} x_j \le |L|.$$

Démonstration :

Etant donné $\underline{x}$ définie par $\underline{x}_j = 1 \;\forall\, j \in L,\; 0 \;\forall\, j \notin L$:

$$\left.\begin{array}{l} a\underline{x} = \sum_{j\in L} a_j \;;(68a) \\ \sum_{j=1}^{n} x_j = |L| \end{array}\right\} \Longrightarrow \underline{x} \in F(K)$$

$$\left.\begin{array}{l} \forall\, x \in V \quad : \sum_{j=1}^{n} x_j > |L| \\ \qquad\qquad (67) \end{array}\right\} \Longrightarrow ax \ge a\underline{x} + \min_{j\notin L} a_j \;\; (68b) \Longrightarrow \dots$$

$$\dots \Longrightarrow ax > b(|L|+1) \ge b\left(\sum_{j=1}^{n} x_j\right) \Longrightarrow x \notin F(K)$$

Note : La solution $\underline{x}$ définie ci-dessus est en conséquence une solution réalisable possédant le maximum de variables non nulles ; l'algorithme B_3 associé déterminera $\underline{x}$ pour poser $u = |L|$. □

Le théorème suivant est également présenté dans [EK9] (mais avec le classement par ordre décroissant de la suite (c_j/a_j), et une démonstration différente) :

Théorème 36

$$\text{Si } \exists\, U \subset \{1,\dots,n\} : \begin{cases} \forall\, j \in U \;\; c_j/a_j \geq c_k/a_k \;\; \forall\, k \notin U & (69) \\ \sum_{j\in U} a_j - a_t < b(|U|) \leq \sum_{j\in U} a_j & (70) \\ \text{avec } t : c_t/a_t = \min_{j\in U} c_j/a_j \end{cases}$$

$$\text{alors } \bar{v}(K \mid \sum_{j=1}^{n} x_j \geq |U|) = \left\lfloor \sum_{j\in U\setminus\{t\}} c_j + c_t\left(b(|U|) - \sum_{j\in U\setminus\{t\}} a_j\right)/a_t \right\rfloor$$

Démonstration : $\forall\, U \subset \{1,\dots,n\}$

$$v(K \mid \sum_{j=1}^{n} x_j \geq |U|) = v(K \mid \sum_{j=1}^{n} x_j > |U| - 1) \text{ puisque } x_j \in \{0,1\} \;\forall\, j$$

$$= \max\{cx \mid ax \leq b(\sum_{j=1}^{n} x_j);\ \sum_{j=1}^{n} x_j > |U|-1, x_j = 0 \text{ ou } 1\ \forall j\}$$

b fonction non croissante; $\sum_{j=1}^{n} x_j \in \mathbb{N}\ \forall\, x \in V$

$$\dots \Longrightarrow v(K \mid \sum_{j=1}^{n} x_j \geq |U|) \leq \max\{cx \mid ax \leq b(|U|);\ \sum_{j=1}^{n} x_j > |U|-1;\ x_j = 0 \text{ ou } 1\ \forall j\}$$

$$\leq \max\{cx \mid ax \leq b(|U|);\ \sum_{j=1}^{n} x_j > |U|-1;\ 0 \leq x_j \leq 1\ \forall j\} \quad (71)$$

Etant donné $U \subset \{1,\dots,n\}$ vérifiant (69) et (70), le problème du knapsack classique :

$$(P_2) \qquad \begin{cases} \max\ cx \\ \text{s.c. } ax \leq b(|U|) \\ x \in [V] \end{cases}$$

admet pour solution réalisable optimale (partie II, chapitre 2) $\bar{x}$ définie par :

$$\bar{x}_j = \begin{cases} 1 & \forall\, j \in U \setminus \{t\} \\ \alpha & j = t \\ 0 & \forall\, j \notin U \end{cases}$$

où $\alpha = (b(|U|) - \sum_{j \in U \setminus \{t\}} a_j) / a_t \in]0,1]$ (d'après (70)).

De plus (P_2) est une relaxation de

$$(P_1) \quad \begin{cases} \max \ cx \\ \text{s.c. } ax \leq b(|U|) \\ \sum_{j=1}^{n} x_j \geq |U| - 1 \\ x \in [V] \end{cases}$$

qui admet $\bar{x}$ pour solution réalisable puisqu'à l'évidence

$$|U| - 1 < \sum_{j=1}^{n} \bar{x}_j \leq |U|$$

donc

$$\left.\begin{array}{l} v(P_1) = v(P_2) = c\bar{x} = \sum_{j \in U \setminus \{t\}} c_j + c_t\, \alpha \\ \\ v(K) \in \mathbf{N} \ ; \ (71) \end{array}\right\} \Longrightarrow \bar{v}(K \mid \sum_{j=1}^{n} x_j \geq |U|) = \lfloor c\bar{x} \rfloor. \ \square$$

Pour tout problème (K) ; considérons en effet l'exemple numérique suivant :

Un ensemble $U \subset \{1,\ldots,n\}$ vérifiant (69) et (70) n'existe pas nécessairement

Problème test 1 de [EK9] (voir Annexe) dans lequel sont seules modifiées les valeurs de b(6) et b(7) de sorte que

$$b(6) = q \leq 43 \qquad \text{et} \qquad q \geq b(7) \geq 29 = b(8)$$

pour conserver la non croissance de la fonction b ;

Les cinq plus grandes valeurs des rapports c_j/a_j $\forall\, j$ sont dans l'ordre :

$c_{11}/a_{11} = 56/4 > c_7/a_7 = 64/7 > c_{12}/a_{12} = 32/6 > c_{16}/a_{16} = 45/14 > c_8/a_8 = 35/21$

Par définition, $b(|U|)$ décroît lorsque $|U|$ croît :

$U = \{11\}$ $\quad s_1 = a_{11} = 4 < b(1) = 264$;
$U = \{11,7\}$ $\quad s_2 = s_1 + a_7 = 11 < b(2) = 223$;
$U = \{11,7,12\}$ $\quad s_3 = s_2 + a_{12} = 17 < b(3) = 216$;
$U = \{11,7,12,16\}$ $\quad s_4 = s_3 + a_{16} = 31 < b(4) = 214$;
$U = \{11,7,12,16,8\}$ $\quad s_5 = s_4 + a_8 = 44 < b(5) = 126$;

mais de plus, par hypothèse, $s_5 > b(6)$

$$\Longrightarrow \left[\begin{array}{l} \forall\, U \subset \{1,\ldots,n\} \text{ vérifiant (69) et } |U| \geq 6 \\ \\ \quad \sum_{j\in U} a_j > b(|U|) \quad \underline{\text{mais}} \quad \sum_{j\in U} a_j - a_t > b(|U|) \end{array} \right.$$

ce qui contredit la double inégalité (70).

Par contre, dans cet exemple a été mis en évidence un ensemble $U = \{11,7,12,16,8\}$ vérifiant : $b(|U| + 1) \leq \sum_{j\in U} a_j < b(|U|)$; le résultat suivant prouve que $\sum_{j\in U} c_j$ est alors une borne supérieure de $v(K \mid \sum_{j=1}^{n} x_j \geq |U| + 1)$:

Théorème 37

$$\textit{Si } \exists\, U \subset \{1,\ldots,n\} \left[\begin{array}{l} \forall\, j \in U \quad c_j/a_j \geq c_k/a_k \quad \forall\, k \notin U \\ \\ b(|U| + 1) \leq \sum_{j\in U} a_j < b(|U|) \end{array} \right. \qquad (72)$$

$$\textit{alors} \qquad \bar{v}(K \mid \sum_{j=1}^{n} x_j \geq |U| + 1) = \sum_{j\in U} c_j$$

Démonstration

Soit $U \subset \{1,\dots,n\}$ vérifiant (69) et (72), la fonction b étant non croissante :

$$v(K \mid \sum_{j=1}^{n} x_j \geq |U|+1) \leq \max \{cx \mid ax \leq b(|U|+1);\ \sum_{j=1}^{n} x_j \geq |U|+1\ ;\ x_j=0 \text{ ou } 1\ \forall\ j\}\ ;$$

de plus, $\sum_{j\in U} a_j \geq b(|U|+1)$ d'après (72), et $\Sigma\, x_j \geq |U|+1 \Longrightarrow \Sigma\, x_j \geq |U|$, donc

$$v(K \mid \sum_{j=1}^{n} x_j \geq |U|+1) \leq \max \{cx \mid ax \leq \sum_{j\in U} a_j;\ \sum_{j=1}^{n} x_j \geq |U|\ ;\ x_j = 0 \text{ ou } 1\ \forall j\} \quad (73)$$

Le problème du knapsack classique

$$(P_4) \quad \begin{cases} \max\ cx \\ \text{s.c. } ax \leq \sum_{j\in U} a_j \\ x \in [V] \end{cases}$$

où U vérifie (69)

admet pour solution optimale réalisable $\bar{x}$ définie par $\bar{x}_j = 1\ \forall j \in U$, $0\ \forall j \notin U$; (P_4) étant, en outre, une relaxation de

$$(P_3) \quad \begin{cases} \max\ cx \\ \text{s.c. } ax \leq \sum_{j\in U} a_j \\ \sum_{j=1}^{n} x_j \geq |U| \\ x \in [V] \end{cases}$$

qui admet $\bar{x}$ pour solution réalisable $(a\bar{x} = \sum_{j\in U} a_j\ ;\ \sum_{j=1}^{n} \bar{x}_j = |U|)$,

$$\left.\begin{array}{l} \text{alors } v(P_3) = v(P_4) = \sum_{j\in U} c_j \\ (73) \end{array}\right\} \Longrightarrow \bar{v}(K \mid \sum_{j=1}^{n} x_j \geq |U|+1) = \sum_{j\in U} c_j. \qquad \square$$

(72) L'algorithme B_4 déterminera $U \subset \{1,\dots,n\}$ vérifiant (69) et (70) ou suivant les problèmes, dans le but d'utiliser le résultat du :

Corollaire

Etant donnés $\underline{v}(K)$ *et* $U \subset \{1,\ldots,n\}$ *vérifiant* (69),

$$\begin{cases} c_t/a_t = \min\limits_{j\in U} c_j/a_j \\ \\ \bar{z} = \begin{cases} \left\lfloor \sum\limits_{j\in U\setminus\{t\}} c_j + c_t(b(|U|) - \sum\limits_{j\in U\setminus\{t\}} a_j)/a_t \right\rfloor & \text{si } U \text{ vérifie (70)} \\ \\ \sum\limits_{j\in U} c_j & \text{si } U \text{ vérifie (72)} \end{cases} \end{cases}$$

$u' = |U| - 1$ (*resp.* $|U|$) *si* U *vérifie* (70) (*resp.* (72)).

si $\bar{z} \le \underline{v}(K)$ *alors*

$\forall\, x \in F(K)$: $cx > \underline{v}(K) \implies \sum\limits_{j=1}^{n} x_j \le u'$.

Démonstration :

$$\left.\begin{array}{l} v(K) = \max\{v(K \mid \sum\limits_{j=1}^{n} x_j \ge u'+1),\ v(K \mid \sum\limits_{j=1}^{n} x_j \le u')\} \\ \text{théorèmes 36 et 37} \Longrightarrow \bar{v}(K \mid \sum\limits_{j=1}^{n} x_j \ge u'+1) = \bar{z} \\ \bar{z} \le \underline{v}(K) \quad \text{par hypothèse} \end{array}\right\} \Longrightarrow \cdots$$

... $\Longrightarrow$ si $\exists$ x réalisable : $cx > \underline{v}(K)$ alors cette solution ne peut être que solution réalisable de $(K \mid \sum\limits_{j=1}^{n} x_j \le u')$, d'où la conclusion. □

III.2 - LINÉARISATIONS DU PROBLÈME - APPLICATIONS :

La construction de l'enveloppe convexe de

$$M_{1,m} = \{p_1, p_2, \ldots, p_m\}$$

permet d'envisager deux types de linéarisation du problème (K) (voir figure

III.2.1 - Linéarisation minorante

Toute droite d'équation $y = \underline{\alpha}s + \underline{\beta}$ séparant l'origine de $[M_{1,m}]$ sera telle que

$$\forall\ s \in \{1,\ldots,m\} \qquad \underline{\alpha}s + \underline{\beta} \leq b(s) \tag{74}$$

et donc le problème du knapsack classique

$$(ILK) \quad \left[\begin{array}{l} \max \ \sum_{j=1}^{n} c_j x_j \\ s.c. \ \sum_{j=1}^{n} (a_j - \underline{\alpha}) x_j \leq \underline{\beta} \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,n \end{array}\right.$$

linéarisation minorante de (K), permettra de construire - par l'intermédiaire d'un algorithme glouton (voir § IV.4) - une borne inférieure de v(K) puisque (74) $\Rightarrow$ v(ILK) $\leq$ v(K).

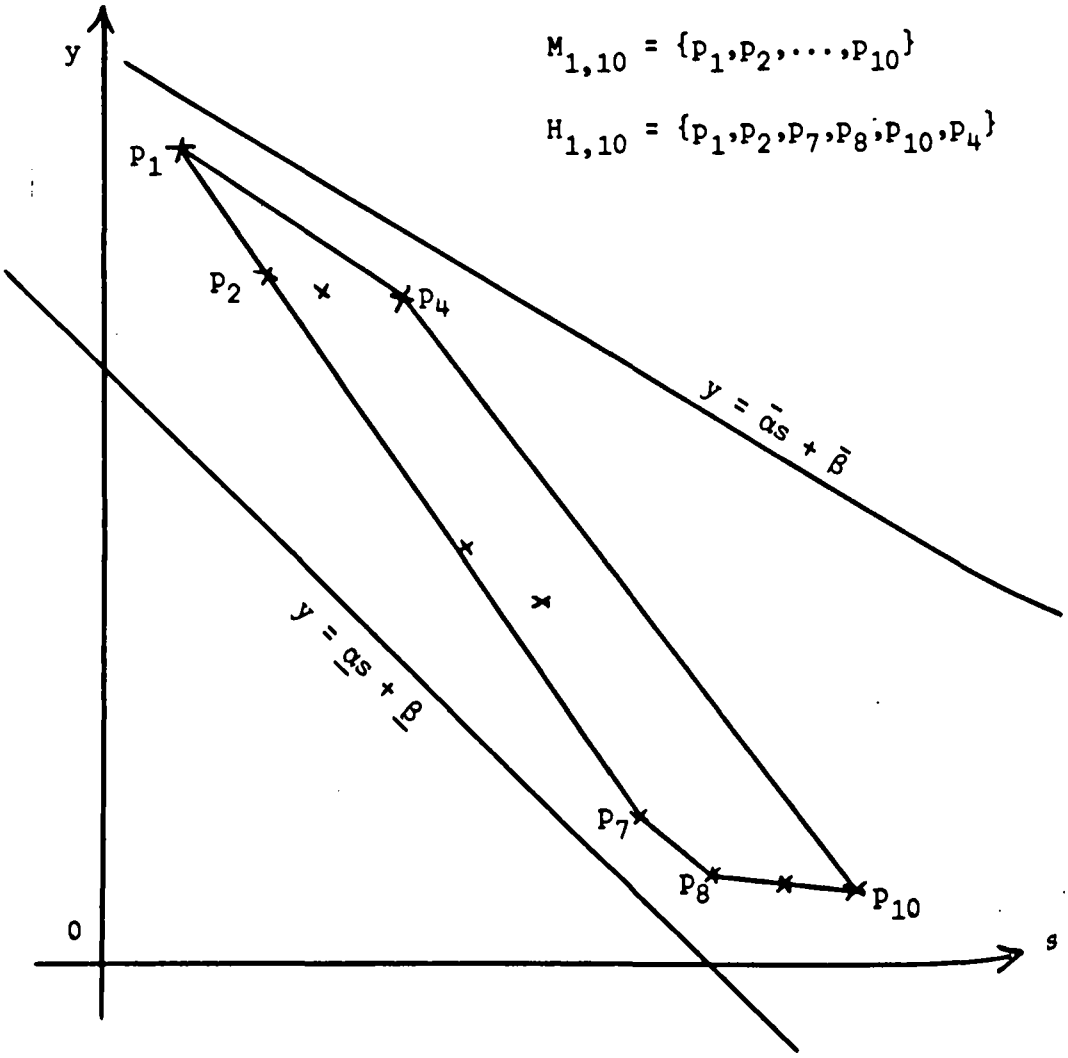


Figure 8

III.2.2 - Linéarisation majorante

De manière tout à fait symétrique, toute droite d'équation $y = \bar{\alpha}s + \bar{\beta}$ "séparée" de l'origine par $[M_{1,m}]$ sera telle que

$$\forall\ s \in \{1,2,\ldots,m\} \qquad b(s) \leq \bar{\alpha}s + \bar{\beta}$$

La résolution de la relaxation du problème du knapsack

$$(\text{ØLK}) \qquad \left[\begin{array}{l} \max \ \sum_{j=1}^{n} c_j x_j \\ \text{s.c.} \ \sum_{j=1}^{n} (a_j - \bar{\alpha}) x_j \leq \bar{\beta} \\ x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,n \end{array}\right.$$

linéarisation majorante de (K), aboutira à une borne supérieure de v(K) (voir §IV.5) et permettra de plus d'éliminer définitivement des variables de (K) en appliquant un algorithme de réduction du type de la partie V du chapitre 2, basé sur le principe :

Pour $j \in \{1,\ldots,n\}$, $\varepsilon \in \{0,1\}$ et $\underline{v}(K)$ donnés :
$\bar{v}(\text{ØLK} \mid x_j = \varepsilon) \leq \underline{v}(K) \Longrightarrow x_j \leftarrow 1 - \varepsilon$.

Remarques :

1°. La monotonie de b fait que les linéarisations minorante et majorante seront toutes de pentes négatives, ce qui implique que les coefficients des contraintes de (ILK) (resp. (ØLK)) seront de la forme $a_j + |\underline{\alpha}|$ (resp. $a_j + |\bar{\alpha}|$) $j=1,\ldots,n$.

2°. Lorsque

$\exists\, \ell, u \in \{1,2,\ldots,m\} : \ell < u$ et $\ell \leq s = \sum_{j=1}^{n} x_j \leq u$

les résultats précédents peuvent être appliqués à $M_{\ell,u}$.

Cas particuliers :

(i) $\underline{b(\ell) > b(u) \text{ et } p_{\ell+1},\ldots,p_{u-1} \in [p_\ell\ p_u]}$: le meilleur choix des linéarisations minorante et majorante s'impose :

$$\underline{\alpha} = \bar{\alpha} = \frac{b(u) - b(\ell)}{u - \ell} \quad \text{et} \quad \underline{\beta} = \bar{\beta} = b(u) - u\underline{\alpha} = b(\ell) - \ell\underline{\alpha}$$

note : $u = \ell+1 \Longrightarrow \underline{\alpha} = \bar{\alpha} = b(u) - b(\ell)$

(ii) $\underline{b(\ell) = b(u)}$: puisque b est non croissante, $b(s) = k$ = constante $\forall s \in \{\ell,\ldots,u\}$; le problème (K) devient :

$$\begin{cases} \max & \sum_{j=1}^{n} c_j\ x_j \\ \text{s.c.} & \sum_{j=1}^{n} a_j\ x_j \leq k \\ & \ell \leq \sum_{j=1}^{n} x_j \leq u \\ & x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,n \end{cases}$$

(iii) $\underline{u = \ell}$: le problème (K) se réduit à

$$\begin{cases} \max & \sum_{j=1}^{n} c_j\ x_j \\ \text{s.c.} & \sum_{j=1}^{n} a_j\ x_j \leq b(u) \\ & \sum_{j=1}^{n} x_j = u \\ & x_j = 0 \text{ ou } 1 \quad j = 1,\ldots,n \end{cases}$$

□

III.2.3 - Remarques sur la perturbation des coefficients de la contrainte

Théorème 38

Etant donnés $j, k \in \{1,\ldots,n\}$, $j \neq k$, *alors*

$$c_j/a_j \geq c_k/a_k \implies c_j/(a_j + \alpha) \geq c_k/(a_k + \alpha)$$

sous l'une des hypothèses exclusives suivantes :

(H_1) $\quad c_j \geq c_k$ *et* $\alpha \in \mathbb{R}_+$

(H_2) $\quad c_j \leq c_k$ *et* $\alpha \in \mathbb{R}_- : \alpha \geq \max(-a_j, -a_k)$

(H_3) $\quad c_j < c_k$ *et* $0 \leq \alpha \leq \alpha^*(a,j,k)$
où $\alpha^*(a,j,k) = (c_j a_k - c_k a_j) / (c_k - c_j)$

(H_4) $\quad c_j > c_k$ *et* $\alpha^*(a,j,k) \leq \alpha \leq 0$.

Démonstration : $c_j/a_j \geq c_k/a_k \implies c_j a_k \geq c_k a_j$ (75)

i)

$$(75) \text{ et } \begin{bmatrix} (H_1) \\ \text{ou} \\ (H_2) \end{bmatrix} \implies c_j a_k + \alpha c_j \geq c_k a_j + \alpha c_k, \text{ soit } c_j (a_k+\alpha) \geq c_k (a_j+\alpha)$$

d'où la conclusion; (il est à noter que $(H_2) \implies a_k+\alpha \geq 0$ et $a_j+\alpha \geq 0$).

ii) Après avoir remarqué que $\begin{bmatrix} (H_3) \implies c_k - c_j > 0 \text{ et } \alpha^*(a,j,k) \geq 0 \\ (H_4) \implies c_k - c_j < 0 \text{ et } \alpha^*(a,j,k) \leq 0 \end{bmatrix}$

$$(75) \text{ et } \begin{bmatrix} (H_3) \\ \text{ou} \\ (H_4) \end{bmatrix} \implies \alpha(c_k - c_j) \leq c_j a_k - c_k a_j \quad \forall\, \alpha \in \begin{bmatrix} [0, \alpha^*(a,j,k)] \text{ sous } (H_3) \\ [\alpha^*(a,j,k), 0] \text{ sous } (H_4) \end{bmatrix}$$

Soit $c_k(\alpha+a_j) \leq c_j(a_k+\alpha)$, d'où la conclusion. □

Le résultat suivant indique dans quelles limites le passage d'une linéarisation à une autre n'apporte pas de perturbation dans la résolution de la relaxation du knapsack par l'algorithme CKR (voir partie II, chapitre 2).

Corollaire

Etant donnés $a'_j = a_j + \alpha_1$ où $\alpha_1 \in \mathbb{R}_+$:

$c_1/a'_1 \geq c_2/a'_2 \geq \ldots\ldots\ldots\ldots \geq c_n/a'_n$,

en posant

$a''_j = a_j + \alpha_2$ *où* $\alpha_2 \in \mathbb{R}_+$, $\alpha_2 \neq \alpha_1$

$\alpha = \alpha_2 - \alpha_1$

et $\bar{\alpha}^*(a',j,k) = \begin{cases} +\infty & \text{sous } (H_1) \\ \alpha^*(a',j,k) & \text{sous } (H_3) \text{ ou } (H_4) \\ \max(-a'_j,-a'_k) & \text{sous } (H_2)\ , \end{cases}$

si $\alpha \in \left]0, \min_{h=1,..,n-1} \bar{\alpha}^*(a',h,h+1)\right]$ *ou* $\alpha \in \left[\max_{h=1,..,n-1} \bar{\alpha}^*(a',h,h+1), \alpha\right[$

alors $c_1/a''_1 \geq c_2/a''_2 \geq \ldots\ldots\ldots \geq c_n/a''_n$

Démonstration :

Il suffit d'appliquer le théorème 38 dans lequel a' remplace a, en notant que $a''_j = a'_j + \alpha$,

$\forall\, h \in \{1,\ldots,n-1\}\ c_h/a'_h \geq c_{h+1}/a'_{h+1} \Longrightarrow c_h/a''_h \geq c_{h+1}/a''_h$

lorsque

$$\left.\begin{array}{lll} \alpha \in \mathbb{R}_+ & \text{si } c_h \geq c_{h+1} & (H_1) \\ \alpha \leq \alpha^*(a',h,h+1) & \text{sinon} & (H_3) \end{array}\right] \quad \text{pour } \alpha > 0 \ ,$$

$$\left.\begin{array}{lll} \alpha \geq \max\,(-a'_h,-a'_{h+1}) & \text{si } c_h \leq c_{h+1} & (H_2) \\ \alpha \geq \alpha^*(a',h,h+1) & \text{sinon} & (H_4) \end{array}\right] \quad \text{pour } \alpha < 0.$$

□

IV - METHODE DE REDUCTION DE LA TAILLE DE (K)

IV.1 - PRINCIPE DE LA METHODE

0 $X_2 \equiv \{1,\ldots,n\}$, $X_0 \equiv X_1 \equiv \emptyset$

1 Détermination de ℓ, $u \in \{1,\ldots,m\}$ tels que

$$\ell \leq \sum_{j \in X_2} x_j \leq u$$

avec (§ IV.2) :

- u : nombre maximum de plus petits éléments de la suite $(a_j)_{j \in X_2}$ dont la somme n'excède pas $b(u)$
- $\ell = r+1$, avec r : nombre maximum de plus grands éléments de la suite $(c_j)_{j \in X_2}$ dont la somme correspondante des a_j n'excède pas $b(r)$.

La détermination de ℓ fournit une valeur initiale de $\underline{v}(K)$.

2 Elimination des variables d'indices j : $a_j > b(\ell)$
(i.e. $X_0 \equiv \{j \mid a_j > b(\ell)\}$ et $X_2 \equiv X_2 \setminus X_0$)

3 Algorithme glouton déterminant $\underline{z}$, minorant de $v(K)$; deux méthodes sont décrites au § IV.4, dont l'une est basée sur des linéarisations minorantes construites à partir de convexes emboîtés contenus dans $[M_{\ell,u}]$ (§IV.3).

4 $\underline{\text{si}}$ $\underline{z} \leq \underline{v}(K)$ $\underline{\text{alors aller en}}$ 6 $\underline{\text{sinon}}$ $\underline{v}(K) \leftarrow \underline{z}$

5 Détermination de ℓ' : nombre minimum de plus grands éléments de la suite $(c_j)_{j \in X_2}$ dont la somme n'est pas inférieure à $\underline{v}(K)+1$ (§ IV.2)

5 si $\ell' > \ell$ alors $X'_0 \equiv \{j \mid a_j > b(\ell')\}$; $X_0 \equiv X_0 \cup X'_0$;

$X_2 \equiv X_2 \setminus X'_0$; $\ell \leftarrow \ell'$

6 Détermination de u' : nombre minimum des plus grands éléments de la suite $(c_j/a_j)_{j\in X_2}$ dont la somme correspondante des a_j n'est pas inférieure à b(u') (§ IV.2) ; cette étape fournit $\underline{z}$, borne inférieure de v(K), et $\bar{z} \geq \max \{cx \mid ax \leq b (\sum_{j\in X_2} x_j)$; $\sum_{j\in X_2} x_j \geq u'$; $x_j = 0$ ou 1 $\forall\, j \in X_2\}$

si $\bar{z} \leq \underline{v}(K)$ et $u' \leq u$ alors $u \leftarrow u' - 1$; aller en 7

si $\underline{z} > \underline{v}(K)$ alors $\underline{v}(K) \leftarrow \underline{z}$

répéter les étapes 4 et 5 avant d'aller en 7

7 Phase de réduction

7.0 élimination des variables d'indices k telles que $a_k > b(\ell) - \sum_{j\in L} a_j = b'(\ell)$
où $|L| = \ell-1$ et $\forall j \in L$, $a_j \leq a_h$ $\forall h \notin L$
(i.e. $X'_0 \equiv \{k \mid a_k > b'(\ell)\}$; $X_0 \equiv X_0 \cup X'_0$; $X_2 \equiv X_2 \setminus X'_0$

7.1 Choix d'une linéarisation majorante de (K) basée sur la sélection d'une droite $y = \bar{\alpha}s + \bar{\beta}$.

7.2 La résolution de la relaxation du knapsack classique
(ØLK) max $\sum_{j\in X_2} c_j x_j$ s.c. $\sum_{j\in X_2} (a_j - \bar{\alpha})\, x_j \leq \bar{\beta}$; $x_j = 0$ ou 1 $\forall\, j \in X_2$

fournit une partition de $X_2 = U \cup \{i\} \cup L$ telle que

$$\begin{cases} \forall\, j \in U,\ c_j/(a_j-\bar{\alpha}) \geq c_i/(a_i-\bar{\alpha}) \geq c_h/(a_h-\bar{\alpha}) \quad \forall\, h \in L \\ \sum_{j\in U} (a_j-\bar{\alpha}) \leq \bar{\beta} < \sum_{j\in U\cup\{i\}} (a_j-\bar{\alpha}) \end{cases}$$

7.3 si $\lfloor v(\overline{\text{ØLK}}) \rfloor \leq \underline{v}(K)$ alors $v(K) \leftarrow \underline{v}(K)$; stop

7.4 si $\underline{v}(\text{ØLK}) > \underline{v}(K)$ et le vecteur associé est une solution réalisable de (K)
alors $\underline{v}(K) \leftarrow \underline{v}(\text{ØLK})$;

7.5 Elimination définitive et fixation à 1 de variables d'indices $j \in U$:

si $\exists\, j \in U : \bar{v}(\text{ØLK} \mid x_j = 0) \leq \underline{v}(K)$ alors $x_j \leftarrow 1$; $X_1 \equiv X_1 \cup \{j\}$

$b(s) \leftarrow b(s) - a_j$, $\forall\, s \in \{\ell, \ldots$

$X_2 \equiv X_2 \setminus \{j\}$;

si $|X_1| = u$ alors $v(K) \leftarrow \underline{v}(K)$; stop

si $|X_1| \geq \ell$ alors $\ell \leftarrow |X_1| + 1$ / $s \leftarrow s - |X_1|$ $s = \ell, \ldots, u$

si $b(\ell) < 0$ alors $v(K) \leftarrow \underline{v}(K)$; stop

si $\exists\, s < u : b(s) \leq 0$ alors $u \leftarrow \max \{s \mid b(s) > 0\}$;

Elimination définitive et fixation à 0 de variables d'indices $j \in L$:

si $\exists\, j \in L : a_j > b(\ell) = \begin{cases} b(\ell) \text{ si } |X_1| \geq \ell - 1 \\ b(\ell) - \sum_{j \in \underline{X}_2} a_j \text{ avec } \begin{cases} |\underline{X}_2| = \ell - |X_1| - 1 \text{ sinon} \\ \forall h \in \underline{X}_2\ a_h \leq a_k\ \forall k \in X_2 \setminus \underline{X}_2 \end{cases} \end{cases}$

ou $\bar{v}(\emptyset LK \mid x_j = 1) \leq \underline{v}(K)$

alors $x_j \leftarrow 0$; $X_0 \equiv X_0 \cup \{j\}$; $X_2 \equiv X_2 \setminus \{j\}$;

Elimination définitive et fixation à 0 de variables d'indices $j \in U$:

si $\exists\, j \in U \cap X_2 : a_j > b'(\ell)$ alors $x_j \leftarrow 0$; $X_0 \equiv X_0 \cup \{j\}$; $X_2 \equiv X_2 \setminus \{j\}$;

si $X_2 = \emptyset$ alors $v(K) \leftarrow \underline{v}(K)$; stop. □

Note

étape 3 : lorsque $\underline{v}(K) = \sum_{j \in U} c_j$ avec $U : \forall\, j \in U,\ c_j \geq c_k,\ \forall\, k \notin U$ déterminé en étape 1, il est clair que $\sum_{j \in U} c_j + \max_{j \notin U} c_j \geq \underline{v}(K) + 1$, d'où $\ell = |U| + 1$; il est donc inutile de dérouler les étapes 4 et 5. □

4.2 - MÉTHODES D'ENCADREMENT DE $\sum_{j \in X_2} x_j$:

Toutes les méthodes exposées ci-dessous sont des généralisations de l'algorithme NKR (chapitre 2) ; elles sont toutes de complexité $O(|X_2|)$ en moyenne (démonstrations analogues à celle de [K13]) ; leur présentation nécessite la donnée de la

Définition 16

Etant donnés $\begin{cases} Z = \bigcup_{j \in Q} \{z_j\} \text{ où } z_j \in \mathbb{R},\ \forall\, j \in Q \subset \mathbb{N} \\ k \in J \subset Q \end{cases}$

la *(Z,k)-partition de J* est définie par la réunion des trois sous-ensembles de J :

$$\begin{cases} U(J,Z,k) = \{j \in J \mid z_j > z_k\} \\ E(J,Z,k) = \{j \in J \mid z_j = z_k\} \\ L(J,Z,k) = J \setminus (U(J,Z,k) \cup E(J,Z,k)). \ \square \end{cases}$$

IV.2.1 - Détermination de $\ell \in \mathbb{N}$: $\sum_{j \in X_2} x_j \geq \ell$

Deux algorithmes justifiés par les théorèmes 33 et 34 du paragraphe III.1 sont présentés :

(i) algorithme B_1 :

Cet algorithme recherche le nombre maximum r de plus grands éléments de la suite $(c_j)_{j \in X_2}$ dont la somme correspondante des a_j n'excède pas b(r), pour poser $\ell = r + 1$.

ALGORITHME B_1

0 $C \equiv \bigcup_{j \in X_2} \{c_j\}$; $U \equiv \emptyset$; $J \equiv X_2$; $r \leftarrow 0$;

1 Sélectionner $k \in J$ et déterminer la (C,k)-partition de J ;

2 si $\sum_{j \in U \cup U(J,C,k)} a_j > b(r+|U(J,C,k)|)$ alors $J \equiv U(J,C,k)$; aller en 1
sinon $U \equiv U \cup U(J,C,k)$;
$r \leftarrow r + |U(J,C,k)|$;

3 si $\sum_{j \in U} a_j = b(r)$ alors $\ell \leftarrow r+1$; stop

4 si $\sum_{j \in U} a_j + \sum_{j \in E(J,C,k)} a_j \geq b(r+|E(J,C,k)|)$ alors construire un sous-ensemble G de E(J,C,k), de cardinalité maximale, tel que $\sum_{j \in U \cup G} a_j \leq b(r + |G|)$; $\ell \leftarrow r + |G| + 1$; stop

5 $U \equiv U \cup E(J,C,k)$; $r \leftarrow r + |E(J,C,k)|$; $J \equiv L(J,C,k)$; aller en 1

Notes :

1- En étape 4, l'ensemble G qui vérifie

$$\sum_{j\in U\cup G} a_j \le b(r+|G|) < \sum_{j\in U\cup G} a_j + a_h \text{ où } h \in E(J,C,k) \setminus G$$

peut être vide lorsque

$$\sum_{j\in U} a_j + a_h > b(r+1) \qquad \forall\, h \in E(J,C,k).$$

2- Dans un but purement pratique - i.e. au niveau du code - il est plus simple d'envisager la définition de la (Z,k)-partition de J suivante :

Définition 16 bis

Etant donnés Z, J et k (voir définition 16), on appellera (Z,k)-partition de J la réunion des trois sous-ensembles de J suivants :

$L(J,Z,k) = \{j\in J-\{k\} \mid z_j \le z_k\}$; $E(J,Z,k)=\{k\}$; $U(J,Z,k)=J\setminus(L(J,Z,k)\cup\{k\})$. □

Avec cette nouvelle définition, l'étape 4 devient :

4' si $\sum_{j\in U} a_j + a_k > b(r+1)$ alors $\ell \leftarrow r+1$; stop

si $\sum_{j\in U} a_j + a_k = b(r+1)$ alors $\ell \leftarrow r+2$; stop

les autres étapes étant inchangées. □

(ii) algorithme B_2 :

Cet algorithme recherche le nombre minimum ℓ de plus grands éléments de la suite $(c_j)_{j\in X_2}$ dont la somme n'est pas inférieure à $\underline{v}(K) + 1$, supposée connue ; il faut remarquer qu'il s'applique à tout problème de maximisation en variables 0-1 à fonction économique linéaire.

ALGORITHME B_2

0 $C \equiv \bigcup_{j \in X_2} \{c_j\}$; $U \equiv \emptyset$; $J \equiv X_2$; $\underline{z} \leftarrow \underline{v}(K) + 1$;

1 Sélectionner $k \in J$ et déterminer la (C,k)-partition de J ;

2 **Si** $\sum_{j \in U \cup U(J,C,k)} c_j > \underline{z}$ **alors** $J \equiv U(J,C,k)$; **aller en** 1

sinon $U \equiv U \cup U(J,C,k)$;

3 **Si** $\sum_{j \in U} c_j = \underline{z}$ **alors** $\ell \leftarrow |U|$; stop

4 **Si** $\sum_{j \in U \cup E(J,C,k)} c_j \geq \underline{z}$ **alors** $\ell \leftarrow |U| + \lceil (\underline{z} - \sum_{j \in U} c_j)/c_k \rceil$; stop

5 $U \equiv U \cup E(J,C,k)$; $J \equiv L(J,C,k)$; **aller en** 1

Notes :

1- En étape 4, $t = \lceil (\underline{z} - \sum_{j \in U} c_j)/c_k \rceil$ est le plus petit entier tel que $\sum_{j \in U} c_j + t * c_k \geq \underline{z}$

2- Avec la définition 16 bis, l'étape 4 est remplacée par

4' **Si** $\sum_{j \in U} c_j + c_k \geq \underline{z}$ **alors** $\ell \leftarrow |U| + 1$; stop

les autres étapes sont inchangées. □

IV.2.2 - Détermination de $u \in \mathbb{N}$: $\sum_{j \in X_2} x_j \leq u$:

Deux algorithmes justifiés par les théorèmes 35, 36, 37 du § III.1 sont présentés :

algorithme B_3 :

Cet algorithme recherche le nombre maximum u de plus petits éléments de la suite $(a_j)_{j\in X_2}$ dont la somme n'excède pas b(u).

ALGORITHME B_3

0 $A \equiv \bigcup_{j\in X_2} \{a_j\}$; $L \equiv \emptyset$; $J \equiv X_2$; $u \leftarrow 0$

1 Sélectionner $k \in J$ et déterminer la (A,k)-partition de J ;

2 si $\sum_{j\in L\cup L(J,A,k)} a_j > b(u + |L(J,A,k)|)$ alors $J \equiv L(J,A,k)$;
aller en 1
sinon $L \equiv L\cup L(J,A,k)$;
$u \leftarrow u+|L(J,A,k)|$;

3 si $\sum_{j\in L} a_j = b(u)$ alors stop

4 si $\sum_{j\in L} a_j + \sum_{j\in E(J,A,k)} a_j \geq b(u+ |E(J,A,k)|)$ alors
$u \leftarrow u + t^*$ avec $t^* = \max \{t \in \mathbb{N} \mid \sum_{j\in L} a_j + t*a_k \leq b(u+t)\}$; stop

5 $L \equiv L \cup E(J,A,k)$; $u \leftarrow u + |E(J,A,k)|$; $J \equiv U(J,A,k)$; aller en 1

Note : Avec la définition 16 bis, l'étape 4 s'écrit :

4' si $\sum_{j\in L} a_j + a_k > b(u+1)$ alors stop

si $\sum_{j\in L} a_j + a_k = b(u+1)$ alors $u \leftarrow u+1$; stop

(ii) algorithme B_4 :

Cet algorithme recherche le nombre minimum u' de plus grands éléments de la suite $(c_j/a_j)_{j\in X_2}$ dont la somme correspondante des a_j n'est pas inférieure à b(u'), dans le but d'obtenir une borne supérieure de $v(K \mid \sum_{j\in X_2} x_j \geq u')$.

Afin de la présenter le plus simplement possible, cette méthode est exposée directement avec la définition 16 bis de la partition :

ALGORITHME B_4

0 $R \equiv \bigcup_{j\in X_2} \{c_j/a_j\}$; $U \equiv \emptyset$; $J \equiv X_2$; $u' \leftarrow z' \leftarrow 0$;

1 Sélectionner $k \in J$ et déterminer la $(R,k)\rightarrow$partition de J ;

2 si $\sum_{j\in U\cup U(J,R,k)} a_j > b(u'+|U(J,R,k)|)$

alors $J \equiv U(J,R,k)$; aller en 1

sinon $U \equiv U \cup U(J,R,k)$; $u' \leftarrow u' + |U(J,R,k)|$;

$z' \leftarrow z' + \sum_{j\in U(J,R,k)} c_j$;

3 si $\sum_{j\in U} a_j = b(u')$ alors stop

sinon $u' \leftarrow u' + 1$;

4 si $\sum_{j\in U} a_j \geq b(u')$ alors stop

5 si $\sum_{j\in U} a_j + a_k \geq b(u')$ alors $z' \leftarrow z' + \lfloor c_k*(b(u')- \sum_{j\in U} a_j)/a_k \rfloor$;

stop

6 $U \equiv U \cup \{k\}$; $z' \leftarrow z' + c_k$; $J \equiv L(J,R,k)$; aller en 1

Remarques : déroulement de la première itération de l'algorithme B_4 :

Etant donnés $J = X_2$, $k \in J$ et la (R,k)-partition de J notée, pour simplifier, (U,E,L), posons $s = \sum_{j \in U} a_j$.

- $s > b(|U|)$ (étape 2) : l'algorithme n'apportant pas d'information sur $\min_{j \in U} c_j/a_j$, il faut itérer le processus sur U ($u' \leq |U|$)

- $s = b(|U|)$ (étape 3) : le théorème 36 permet de conclure que $u' = |U|$

- $b(|U|+1) \leq s < b(|U|)$ (étape 4) : le théorème 37 implique $u' = |U|+1$

- $s < b(|U|+1) \leq s + a_k$ (étape 5) : par définition de la (R,k)-partition, $c_k/a_k \leq c_j/a_j \quad \forall\, j \in U$ d'où la conclusion apportée par le théorème 36 : $u' = |U| + 1$

- $s + a_k < b(|U|+1)$ (étape 6) : il faut poursuivre le processus sur L, en sachant donc que $u' \geq |U|+1$.

IV.3 - DÉTERMINATION DES POINTS EXTREMES DE L'ENVELOPPE CONVEXE D'UN ENSEMBLE DE POINTS DE Z^2

Deux méthodes de détermination de

$$H_{\ell,u} = \{p_{j_1}, p_{j_2}, \ldots, p_{j_r}\},$$

séquence des points extrêmes de l'enveloppe convexe de l'ensemble à $t = u - \ell + 1$ points

$$M_{\ell,u} = \{p_\ell, p_{\ell+1}, \ldots, p_u\} \text{ où } \ell, u \in \mathbb{N},\ \ell < u$$

sont décrites dans ce paragraphe

Pour chacune de ces méthodes appelées respectivement "séquentielle" et "tangentielle", deux variantes sont présentées, l'une de complexité $0(t^2)$ et l'autre dite "dynamique" de complexité $0(t)$.

Note : La notation des éléments de $M_{\ell,u}$ est celle définissant les points de coordonnées $(s,b(s))$, $s \in \{\ell,\ell+1,\ldots,u\}$, mais il est à noter que *l'hypothèse de non croissance de la fonction* b, nécessaire à la définition du problème (K), *n'est pas retenue dans les méthodes présentées* qui aboutissent donc au résultat cherché même si $\exists\ s_1, s_2 \in \{\ell,\ldots,u\} : s_1 < s_2 \Longrightarrow b(s_1) < b(s_2)$. □

IV.3.1 - Notations et résultats préliminaires :

Propriété 9

(i) $p_\ell,\ p_u \in H_{\ell,u}$

(ii) $\forall\ p_s \in M_{\ell,u} : p_s \in [p_\ell\ p_u] \Longrightarrow p_s \notin H_{\ell,u}$

Démonstration :

(i) p_ℓ et p_u, points d'abscisses respectivement minimale et maximale de $M_{\ell,u}$, font nécessairement partie de $H_{\ell,u}$ par définition d'un point extrême. Il est à remarquer que dans le cas particulier où b est non croissante, ces deux points sont également d'ordonnées respectivement maximale et minimale.

(ii) Une conséquence directe du point (i) est que $[p_\ell\ \ p_u] \in [M_{\ell,u}]$, d'où la conclusion. □

* Notations :

$$\forall\ j, k \in \{\ell,\ldots,u\} : j < k,\ \theta_{j,k} = \frac{b(k) - b(j)}{k - j}$$

désignera la pente de la droite $(p_j\ p_k)$.

On posera en outre

$$\bar{M}_{\ell,u} = \{p_j \in M_{\ell,u} \mid \theta_{\ell,j} > \theta_{\ell,u}\} \cup \{p_\ell, p_u\}$$

$$\underline{M}_{\ell,u} = \{p_j \in M_{\ell,u} \mid \theta_{\ell,j} < \theta_{\ell,u}\} \cup \{p_\ell, p_u\}$$

et $\bar{H}_{\ell,u}$ (resp. $\underline{H}_{\ell,u}$) la séquence des points extrêmes de $[\bar{M}_{\ell,u}]$ (resp. $[\underline{M}_{\ell,u}]$).

Propriété 10

(i) $H_{\ell,u} = \bar{H}_{\ell,u} \cup \underline{H}_{\ell,u}$

(ii) $\bar{H}_{\ell,u} \cap \underline{H}_{\ell,u} = \{p_\ell, p_u\}$

Démonstration :

La propriété 9 implique

$$[M_{\ell,u}] = [\bar{M}_{\ell,u}] \cup [\underline{M}_{\ell,u}]$$

et $[\bar{M}_{\ell,u}] \cap [\underline{M}_{\ell,u}] = [p_\ell\ p_u]$, d'où la conclusion. □

* Principes des méthodes :

(i) La méthode "séquentielle" est basée sur la *décomposition* du convexe en *triangles* possédant tous un sommet commun : l'un ou l'autre des points p_ℓ et p_u (voir figure 9).

Si le choix s'est porté sur p_u, la construction de $H_{\ell,u}$ se fera en réunissant les points extrêmes de $[\bar{M}_{\ell,u}]$ et de $[\underline{M}_{\ell,u}]$ (propriété 10) de la manière suivante :

En notant :

$\bar{H}_{\ell,u} = \{p_\ell, p_{i_1}, \ldots, p_{i_{r_1}}, p_u\}$, $\underline{H}_{\ell,u} = \{p_\ell, p_{h_1}, \ldots, p_{h_{r_2}}, p_u\}$

avec $\ell < i_1 < \ldots < i_{r_1} < u$, $\ell < h_1 < \ldots < h_{r_2} < u$ et $r_1 + r_2 = r-2$

$T(s,j,k)$ le triangle de sommets p_s, p_j et p_k,

une décomposition du convexe sera

$$[\bar{M}_{\ell,u}] = T(\ell,i_1,u) \cup \left(\bigcup_{k=1}^{r_1-1} T(i_k,i_{k+1},u) \right)$$

$$[\underline{M}_{\ell,u}] = T(\ell,h_1,u) \cup \left(\bigcup_{k=1}^{r_2-1} T(h_k,h_{k+1},u) \right).$$

Les triangles étant construits dans l'ordre croissant de k :

i.e., pour $[\bar{M}_{\ell,u}]$, dans l'ordre

$T(\ell,i_1,u)$, $T(i_1,i_2,u)$,..., $T(i_{r_1-1},i_{r_1},u)$

et, pour $[\underline{M}_{\ell,u}]$, dans l'ordre

$T(\ell,h_1,u)$, $T(h_1,h_2,u)$,..., $T(h_{r_2-1},h_{r_2},u)$

les éléments de $\bar{H}_{\ell,u}$ et de $\underline{H}_{\ell,u}$ seront déterminés de manière *séquentielle* dans l'ordre croissant des indices.

(ii) La méthode "tangentielle" est basée sur la *décomposition* du convexe *en quadrilatères convexes* superposés auxquels il faudra adjoindre un (resp. deux) triangle(s) si r_1 ou r_2 est impair (resp. r_1 et r_2 sont impairs) (voir figure 10) :

Avec les mêmes notations que dans la partie (i), on posera de plus $Q(s,i,j,k)$ le quadrilatère convexe de sommets p_s, p_i, p_j et p_k pour aboutir à une décomposition du type

$$[\bar{M}_{\ell,u}] = Q(\ell,i_1,i_{r_1},u) \cup \left(\bigcup_{k=2}^{\bar{\gamma}} Q(i_{k-1},i_k,i_{r_1-k+1},i_{r_1-k+2}) \right) \cup \bar{T}$$

avec
- $\bar{\gamma} = r_1/2$ et $\bar{T} = \emptyset$ si r_1 est pair
- $\bar{\gamma} = \lfloor r_1/2 \rfloor$ et $\bar{T} = T(i_{\bar{\gamma}}, i_{\bar{\gamma}+1}, i_{\bar{\gamma}+2})$ sinon

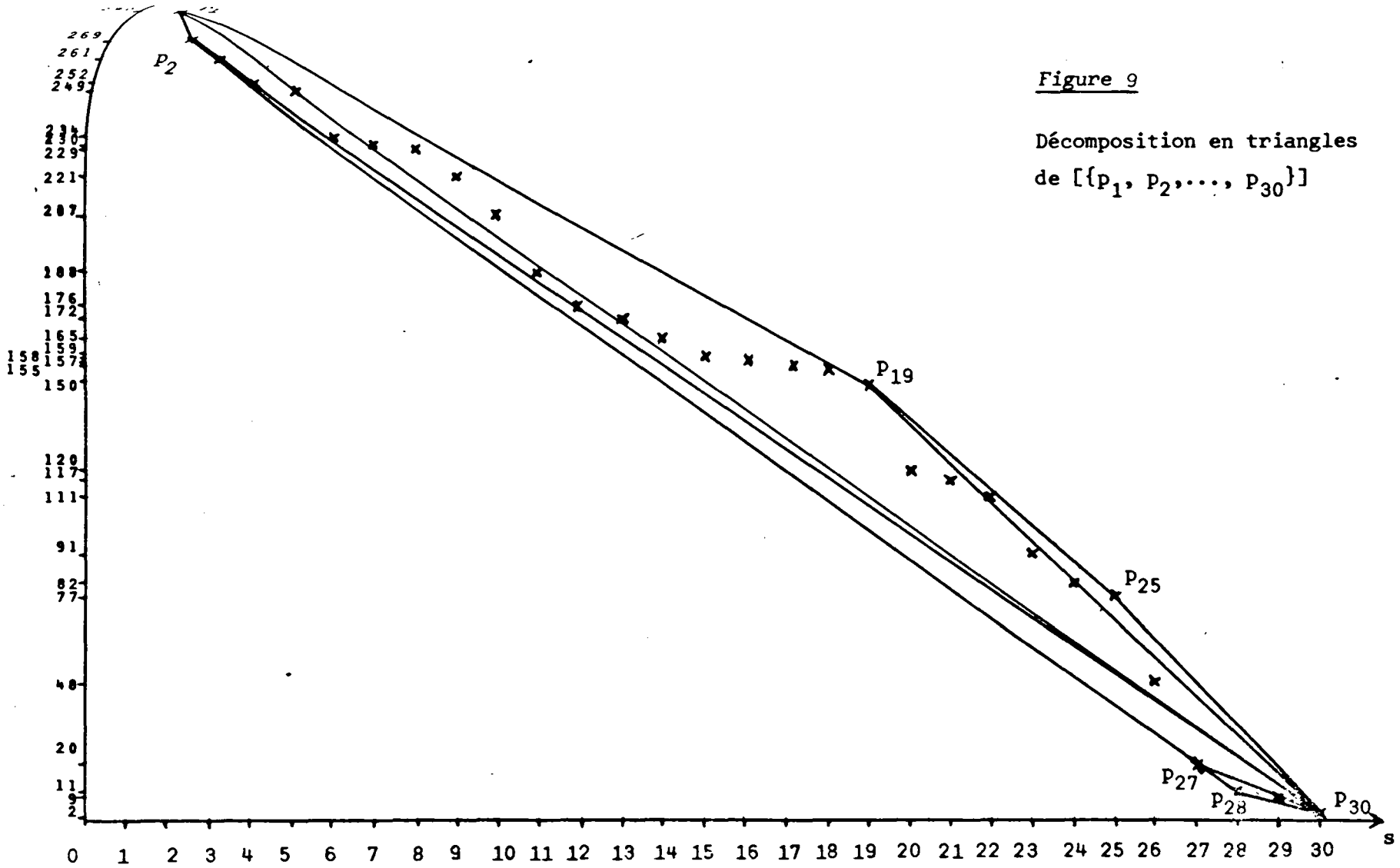


Figure 9

Décomposition en triangles de $[\{p_1, p_2, \ldots, p_{30}\}]$

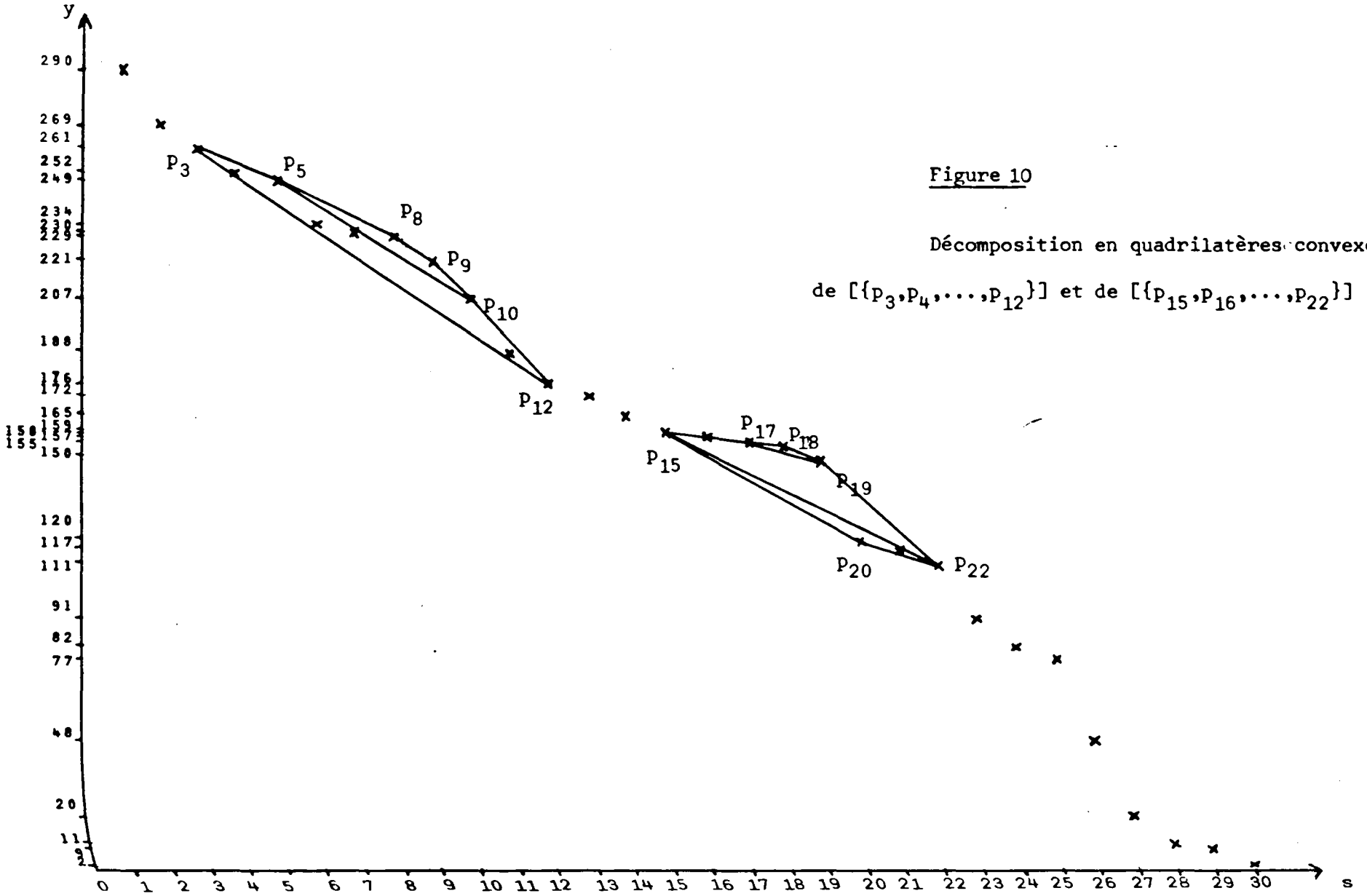


Figure 10

Décomposition en quadrilatères convexes de $[\{p_3,p_4,\ldots,p_{12}\}]$ et de $[\{p_{15},p_{16},\ldots,p_{22}\}]$

$$[M_{\ell,u}] = Q(\ell,h_1,h_{r_2},u) \cup \left(\bigcup_{k=2}^{\Upsilon} Q(h_{k-1},h_k,h_{r_2-k+1},h_{r_2-k+2})\right) \cup \underline{T}$$

avec
$$\begin{cases} \Upsilon = r_2/2 \quad \text{et} \quad \underline{T} = \emptyset & \text{si } r_2 \text{ est pair} \\ \Upsilon = \lfloor r_2/2 \rfloor \quad \text{et} \quad \underline{T} = T(i_\Upsilon, i_{\Upsilon+1}, i_{\Upsilon+2}) & \text{sinon} \end{cases}$$

Les quadrilatères étant construits dans l'ordre croissant de k les triangles $\bar{T}$ ou $\underline{T}$ (s'ils existent) sont introduits en dernier lieu - les éléments de $\bar{H}_{\ell,u}$ et de $\underline{H}_{\ell,u}$ seront déterminés de manière *alternée* (par exemple, pour $\bar{H}_{\ell,u}$, dans l'ordre $p_{i_1}, p_{i_{r_1}}, p_{i_2}, p_{i_{r_1-1}}, p_{i_3}, p_{i_{r_1-2}}$, ...etc.).

IV.3.2 - Méthode séquentielle

(1) Variante 1 : (algorithme S_1) :

En posant, pour $j \in \{\ell,\ldots,u-1\}$ donné,

$$\bar{M}^j_{\ell,u} = \{p_k \in \bar{H}_{\ell,u} \mid k > j\}$$

et $$\underline{M}^j_{\ell,u} = \{p_k \in \underline{H}_{\ell,u} \mid k > j\},$$

l'algorithme S_1 déterminant les éléments de $\bar{H}_{\ell,u}$ et $\underline{H}_{\ell,u}$ est le suivant :

CONSTRUCTION DE $\bar{H}_{\ell,u}$

0 $\bar{H}_{\ell,u} \equiv \{p_\ell\}$, $i_0 \leftarrow \ell$, $k \leftarrow 0$

1 $\theta_{i_k,w^*} = \max \{\theta_{i_k,w} \mid p_w \in \bar{M}^{i_k}_{\ell,u}\}$

2 $k \leftarrow k + 1$, $i_k \leftarrow w^*$

$\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \cup \{p_{i_k}\}$

3 si $i_k \neq u$ alors aller en 1

4 stop

CONSTRUCTION DE $\underline{H}_{\ell,u}$

0 $\underline{H}_{\ell,u} \equiv \{p_\ell\}$, $h_0 \leftarrow \ell$, $k \leftarrow 0$

1 $\theta_{h_k,w^*} = \min \{\theta_{h_k,w} \mid p_w \in \underline{M}^{h_k}_{\ell,u}\}$

2 $k \leftarrow k + 1$, $h_k \leftarrow w^*$
$\underline{H}_{\ell,u} \equiv \underline{H}_{\ell,u} \cup \{p_{h_k}\}$

3 si $h_k \neq u$ alors aller en 1

4 stop

note : Cet algorithme sera plus efficace si $\bar{M}_{i_k,\,u} \setminus \{p_{i_k}, p_u\}$ (resp. $\underline{M}_{h_k,\,u} \setminus \{p_{h_k}, p_u\}$) remplace $\bar{M}^{i_k}_{\ell,u}$ (resp. $\underline{M}^{h_k}_{\ell,u}$)

Théorème 39

En posant $\bar{t} = |\bar{M}_{\ell,u}| - 1$ *et* $\underline{t} = |\underline{M}_{\ell,u}| - 1$, *l'algorithme* S_1 *est de complexité* $O(\bar{t}^2+\underline{t}^2)$ *dans le cas le plus défavorable*

Démonstration

Le plus mauvais cas pour l'algorithme S_1 correspond à un ensemble $M_{\ell,u}$ tel que

$$\begin{cases} |\bar{M}^k_{\ell,u}| = \bar{t} - k & \forall\, p_k \in \bar{M}_{\ell,u} \\ |\underline{M}^k_{\ell,u}| = \underline{t} - k & \forall\, p_k \in \underline{M}_{\ell,u} \end{cases}$$

Pour un tel cas, les $\bar{t} + \underline{t}$ itérations nécessaires à la détermination de $\bar{H}_{\ell,u}$ et $\underline{H}_{\ell,u}$, correspondent à un nombre global d'opérations borné par

$k\,[(\bar{t} + (\bar{t}-1) + \ldots + 1) + (\underline{t} + (\underline{t}-1) + \ldots + 1)]$ où $k \in \mathbb{N}$

d'où la conclusion. □

Remarque : l'algorithme S_1 est l'une des deux méthodes proposées par Chassery [G9] ; le principe de l'autre méthode - dite itérative - est le suivant (dans le cas de la construction de $\bar{H}_{\ell,u}$) :

En posant $\forall\, j, k \in [\ell,u]$ $j < k$: $\bar{M}^*_{j,k} = \bar{M}_{j,k} \setminus \{p_j, p_k\}$

0 $S \equiv \bar{M}^*_{\ell,u}$; $\underline{\ell} \leftarrow \ell$; $\underline{u} \leftarrow u$; $k \leftarrow 0$; Liste $\equiv \emptyset$; $\bar{H}_{\ell,u} \equiv \{p_\ell, p_u\}$

1 Si $S = \emptyset$ alors aller en 5

2 $d(p_w{}^*, [p_{\underline{\ell}} p_{\underline{u}}]) = \max \{d(p_w, [p_{\underline{\ell}}\ p_{\underline{u}}]) \mid p_w \in S\}$

3 $\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \cup \{p_w{}^*\}$

$\underline{4}$ $\quad S \equiv \bar{M}^{*}_{\underline{\ell} w^{*}}$

$\quad k \leftarrow k+1 \; ; \; L_k \equiv \bar{M}^{*}_{w^{*},\underline{u}} \; ; \; \text{Liste} \equiv \text{Liste} \cup \{L_k\}$

$\quad \underline{\text{aller en}}\ 1$

$\underline{5}$ $\quad \underline{\text{Si}}\ k = 0\ \underline{\text{alors}}\ \text{stop}$

$\quad\quad\quad \underline{\text{sinon}}\ S \equiv L_k \; ; \; k \leftarrow k-1$

$\quad\quad\quad\quad \text{Liste} \equiv \text{Liste} \setminus \{S\} \; ; \; \underline{\text{aller en}}\ 1.$

(ii) Variante 2 (algorithme S_2) :

Etant donnés

$$\bar{M}_{\ell,u} = \{p_\ell, p_{\bar{i}_1}, p_{\bar{i}_2}, \ldots, p_{\bar{i}_{\bar{r}}}, p_u\} \; ; \; \bar{H}^{*}_{\ell,j} = \bar{H}_{\ell,j} \cup \{p_u\} \qquad \forall\, j \neq u$$

$$\underline{M}_{\ell,u} = \{p_\ell, p_{\underline{h}_1}, p_{\underline{h}_2}, \ldots, p_{\underline{h}_{\underline{r}}}, p_u\} \; ; \; \underline{H}^{*}_{\ell,j} = \underline{H}_{\ell,j} \cup \{p_u\}$$

la variante dite "dynamique" de la méthode séquentielle est basée sur le principe suivant

en posant $\bar{i}_0 = \underline{h}_0 = \ell$ et $\bar{i}_{\bar{r}+1} = \underline{h}_{\underline{r}+1} = u$,

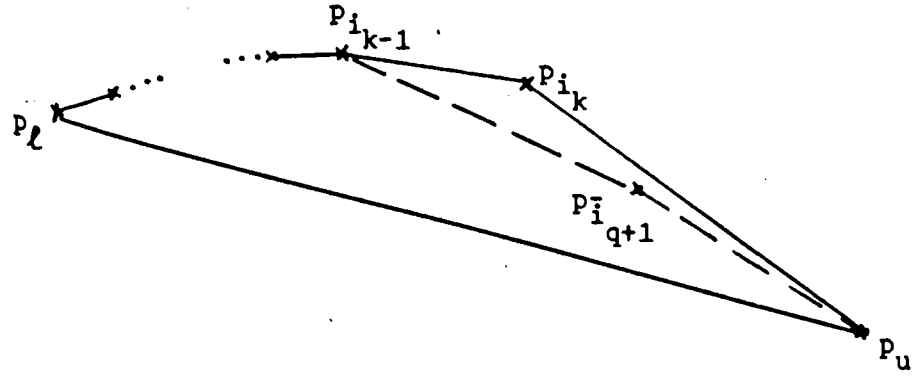
$\bar{H}^{*}_{\ell,\bar{I}_{q+1}}$ est déterminé en fonction de $\bar{H}^{*}_{\ell,\bar{I}_q}$ pour $q = 1,2,\ldots,\bar{r}$

$\underline{H}^{*}_{\ell,\underline{h}_{q+1}}$ est déterminé en fonction de $\underline{H}^{*}_{\ell,\underline{h}_q}$ pour $q = 1,2,\ldots,\underline{r}$

en utilisant les règles suivantes présentées uniquement dans le cadre de $\bar{H}_{\ell,u}$ (il suffit de changer les sens des inégalités pour $\underline{H}_{\ell,u}$) :

Etant donnés $\bar{H}^{*}_{\ell,i_q} = \{p_\ell, p_{i_1}, \ldots, p_{i_k}\}$ et $p_{\bar{I}_{q+1}}$ le nouveau point de $\bar{M}_{\ell,u}$ envisagé :

Règle 1 : $p_{\bar{i}_{q+1}}$ n'est pas point extrême de $[\bar{M}_{\ell,\bar{i}_{q+1}} \cup \{p_u\}]$:

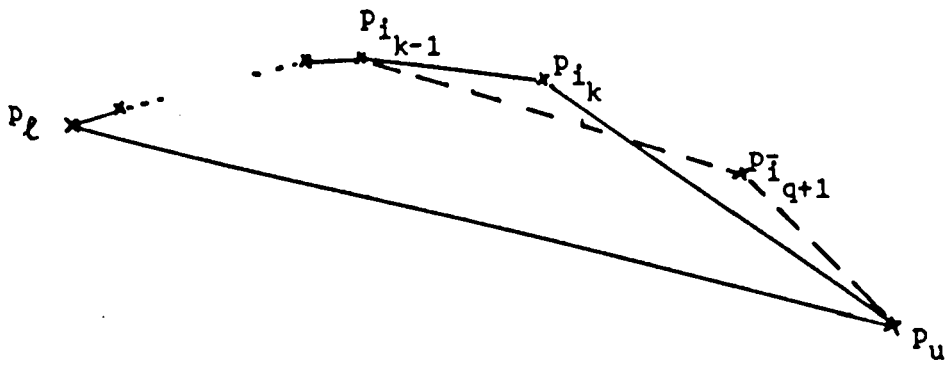


si $\theta_{i_{k-1}i_k} > \theta_{i_{k-1}\bar{i}_{q+1}}$ et $\theta_{i_k u} \leq \theta_{\bar{i}_{q+1}u}$ (76)

alors $\bar{H}^*_{\ell,\bar{i}_{q+1}} = \bar{H}^*_{\ell,\bar{i}_q}$. □

Note : en utilisant la droite $(p_{i_k} p_u)$ - d'équation $y = \theta_{i_k u}(s-u) + b(u)$ - les relations (76) sont équivalentes à $b(\bar{I}_{q+1}) \leq \theta_{i_k u}(\bar{i}_{q+1}-u) + b(u)$. □

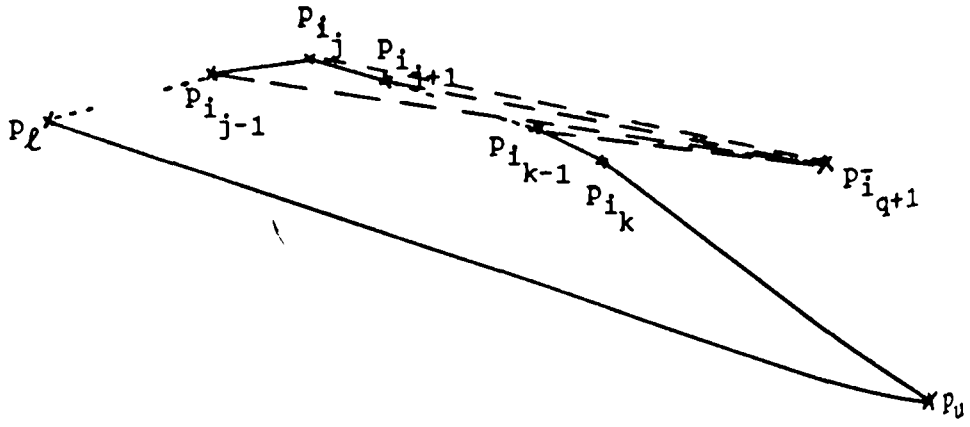
Règle 2 : $p_{\bar{i}_{q+1}}$ est point extrême de $[\bar{M}_{\ell,\bar{i}_{q+1}} \cup \{p_u\}]$, introduit sans suppression d'éléments de $\bar{H}^*_{\ell,\bar{i}_q}$:



si $\theta_{i_{k-1}i_k} > \theta_{i_{k-1}\bar{i}_{q+1}}$ et $\theta_{i_k u} > \theta_{\bar{i}_{q+1}u}$

alors $\bar{H}^*_{\ell,\bar{i}_{q+1}} = \bar{H}^*_{\ell,\bar{i}_q} \cup \{p_{\bar{i}_{q+1}}\}$. □

règle 3 : $p_{\bar{i}_{q+1}}$ est point extrême de $[\bar{M}_{\ell,\bar{i}_{q+1}} \cup \{p_u\}]$, introduit avec suppression d'éléments de $\bar{H}^*_{\ell,\bar{i}_q}$:



si $\Theta_{i_w i_{w+1}} \leq \Theta_{i_w \bar{i}_{q+1}}$ $\quad w = k-1, k-2, \ldots, j+1, j$

et $\Theta_{i_{j-1} i_j} > \Theta_{i_{j-1} \bar{i}_{q+1}}$

alors $\bar{H}^*_{\ell,\bar{i}_{q+1}} = (\bar{H}^*_{\ell,\bar{i}_q} \setminus \{p_{i_{j+1}}, \ldots, p_{i_{k-1}}, p_{i_k}\}) \cup \{p_{\bar{i}_{q+1}}\}$. □

ALGORITHME S_2

(construction de $\bar{H}_{\ell,u}$)

0 $\bar{H}_{\ell,u} \equiv \{p_\ell, p_{\bar{i}_1}, p_u\}$; $i_0 \leftarrow \ell$; $i_1 \leftarrow \bar{i}_1$; $k \leftarrow q \leftarrow 1$

1 si $q = \bar{r}$ alors stop

2 $q \leftarrow q + 1$
si $\Theta_{i_{k-1} i_k} > \Theta_{i_{k-1} \bar{i}_q}$
alors si $\Theta_{i_k u} \leq \Theta_{\bar{i}_q u}$ alors aller en 1
sinon $k \leftarrow k+1$; $i_k \leftarrow \bar{i}_q$
$\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \cup \{p_{i_k}\}$
aller en 1

3 $\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \setminus \{p_{i_k}\}$
si $k = 1$ alors aller en 1
$k \leftarrow k-1$
si $\Theta_{i_{k-1} i_k} \leq \Theta_{i_{k-1} \bar{i}_q}$ alors aller en 3
sinon $k \leftarrow k+1$; $i_k \leftarrow \bar{i}_q$
$\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \cup \{p_{i_k}\}$
aller en 1

Théorème 40

En posant $\bar{\bar{t}} = |\bar{M}_{\ell,u}| - 2$ *et* $\underline{t} = |\underline{M}_{\ell,u}| - 2$ *(on suppose que* $\bar{M}_{\ell,u} \not\supseteq \{p_\ell, p_u\}$ *et* $\underline{M}_{\ell,u} \not\supseteq \{p_\ell, p_u\}$*) l'algorithme* S_2 *est de complexité* $O(\bar{\bar{t}} + \underline{t})$.

Démonstration :

Le cas le plus défavorable correspond à un ensemble de points tels que chacun d'eux soit introduit dans une séquence partielle puis supprimé au cours d'une itération ultérieure (cette suppression ne pouvant avoir lieu pour le dernier point de $\bar{M}_{\ell,u} \setminus \{p_\ell, p_u\}$ - ou de $\underline{M}_{\ell,u} \setminus \{p_\ell, p_u\}$ - introduit). Ainsi, pour chaque point, cinq calculs du type $\theta_{jk} = (b(j)-b(k))/(j-k)$ - avec cinq couples (j,k) tous différents - seront à envisager soit 5 divisions et 10 soustractions ; d'où la conclusion. □

IV.3.3 - Méthode tangentielle

Les deux variantes T_1 et T_2 de la méthode tangentielle ont des complexités identiques à celles de S_1 et S_2 respectivement ; les méthodes S_1 et S_2 d'une part, T_1 et T_2 d'autre part, diffèrent par leur mode exploratoire de $M_{\ell,u}$ (séquentielle pour S_1 et S_2, alternée pour T_1 et T_2) ; l'exploration alternée est en outre exploitée pour l'élaboration de règles plus fortes d'exclusion (basées dorénavant sur des quadrilatères convexes) :

(i) Variante 1 (algorithme T_1) :

Sa présentation ne sera faite que dans le cadre de la construction de $\bar{H}_{\ell,u}$ (pour le cas de $\underline{H}_{\ell,u}$; il suffit d'échanger les rôles des minimisations et des maximisations, en ayant pour référence $\underline{M}_{\ell,u}$).

En posant, pour α, $\omega \in \{\ell+1,\ldots,u-1\}$ donnés,

$$\bar{M}^*_{\alpha,\omega} = \bar{M}_{\alpha,\omega} \setminus \{p_\alpha, p_\omega\}$$

l'algorithme T_1 déterminant les éléments de $\bar{H}_{\ell,u}$ est le suivant :

ALGORITHME T_1

(construction de $\bar{H}_{\ell,u}$)

0 $\bar{H}_{\ell,u} \equiv \{p_\ell, p_u\}$; $k_1 \leftarrow \ell$; $k_2 \leftarrow u$

1 si $\bar{M}^*_{k_1,k_2} = \emptyset$ alors stop

2 $\Theta_{k_1,w^*} = \max \{\Theta_{k_1,w} \mid p_w \in \bar{M}^*_{k_1,k_2}\}$

$k_1 \leftarrow w^*$; $\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \cup \{p_{k_1}\}$

3 si $\bar{M}^*_{k_1,k_2} = \emptyset$ alors stop

4 $\Theta_{k_2,w^*} = \min \{\Theta_{k_2,w} \mid p_w \in \bar{M}^*_{k_1,k_2}\}$

$k_2 \leftarrow w^*$; $\bar{H}_{\ell,u} \equiv \bar{H}_{\ell,u} \cup \{p_{k_2}\}$; aller en 1

(ii) Variante 2 (algorithme T_2) :

Seules seront exposées les règles de la variante dite "dynamique" de la méthode tangentielle dont le déroulement est une simple adaptation de celui de l'algorithme S_2 au cas de l'exploration alternée ; leur présentation est faite dans le cadre de la construction de $\bar{H}_{\ell,u}$:

Etant donnés

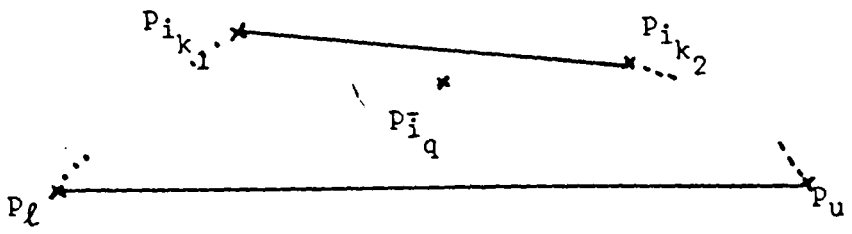
$\bar{H}^{i_{k_2},u}_{\ell,i_{k_1}}$ séquence des points extrêmes de $\bar{M}_{\ell,i_{k_1}} \cup \bar{M}_{i_{k_2},u}$

avec $k_1 < k_2$ et $i_{k_1}, i_{k_2} \in \{\ell,\ldots,u\}$

$p_{\bar{i}_q}$ ($\bar{i}_q \in]i_{k_1}, i_{k_2}[$) le nouveau point de $\bar{M}_{\ell,u}$ envisagé pour construire $\bar{H}^{i_{k_2},u}_{\ell,i_{k_1},\bar{i}_q}$ désignant l'un ou l'autre des

ensembles $\bar{H}_{\ell,i_{k_1}}^{\bar{i}_q,u}$ et $\bar{H}_{\ell,\bar{i}_q}^{i_{k_2},u}$ suivant la place de $p_{\bar{i}_q}$ par rapport à $p_{i_{k_1}}$ et $p_{i_{k_2}}$

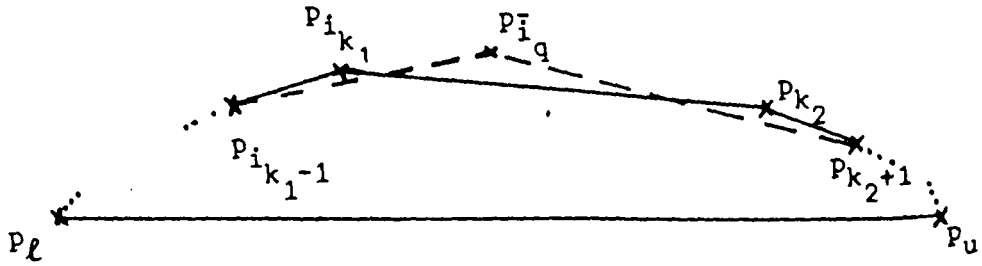
règle 1 : $p_{\bar{i}_q}$ n'est pas point extrême de $\bar{H}_{\ell,i_{k_1},\bar{i}_q}^{i_{k_2},u}$



si $b(\bar{i}_q) \leq \Theta_{i_{k_1},i_{k_2}}(\bar{i}_q - i_{k_1}) + b(i_{k_1})$

alors $\bar{H}_{\ell,i_{k_1},\bar{i}_q}^{i_{k_2},u} = \bar{H}_{\ell,i_{k_1}}^{i_{k_2},u}$. □

règle 2 : $p_{\bar{i}_q}$, point extrême de $\bar{H}_{\ell,i_{k_1},\bar{i}_q}^{i_{k_2},u}$, est introduit sans suppression d'éléments de $\bar{H}_{\ell,i_{k_1}}^{i_{k_2},u}$:

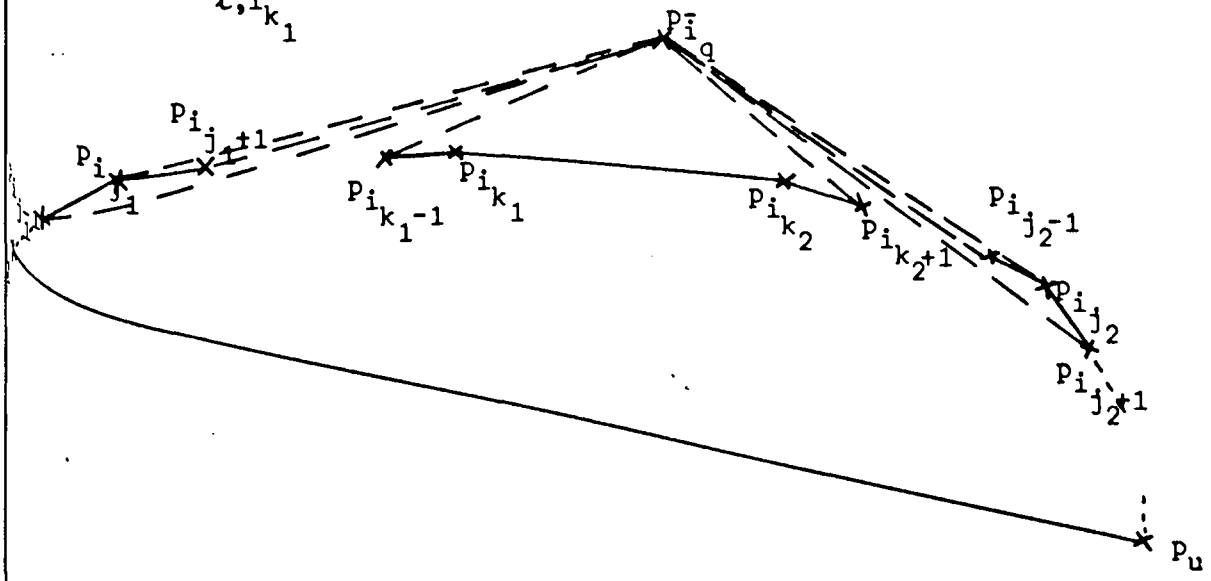


si $b(\bar{i}_q) > \Theta_{i_{k_1},i_{k_2}}(\bar{i}_q - i_{k_1}) + b(i_{k_1})$

et $\Theta_{i_{k_1}-1,i_{k_1}} > \Theta_{i_{k_1}-1,\bar{i}_q}$; $\Theta_{\bar{i}_q,i_{k_2}+1} > \Theta_{i_{k_2},i_{k_2}+1}$

alors $\bar{H}_{\ell,i_{k_1},\bar{i}_q}^{i_{k_2},u} = \bar{H}_{\ell,i_{k_1}}^{i_{k_2},u} \cup \{p_{\bar{i}_q}\}$. □

Règle 3 : $p_{\bar{i}_q}$, point extrême de $\bar{H}^{i_{k_2},u}_{\ell,i_{k_1},\bar{i}_q}$, est introduit avec suppression d'éléments de $\bar{H}^{i_{k_2},u}_{\ell,i_{k_1}}$:



$$\begin{cases} \Theta_{w,w+1} \leq \Theta_{w,\bar{i}_q} & w = i_{k_1-1},\ i_{k_1-2},\ldots,i_{j_1+1},i_{j_1} \\ \Theta_{i_{j_1-1},i_{j_1}} > \Theta_{i_{j_1-1},\bar{i}_q} \end{cases}$$

et

$$\begin{cases} \Theta_{w,w-1} \geq \Theta_{w,\bar{i}_q} & w = i_{k_2+1},\ i_{k_2+2},\ldots,i_{j_2-1},i_{j_2} \\ \Theta_{i_{j_2+1},i_{j_2}} < \Theta_{i_{j_2+1},\bar{i}_q} \end{cases}$$

$$\Rightarrow \bar{H}^{i_{k_2},u}_{\ell,i_{k_1},\bar{i}_q} = (\bar{H}^{i_{k_2},u}_{\ell,i_{k_1}} \setminus \{p_{i_{j_1+1}},\ldots,p_{i_{k_1-1}},p_{i_{k_1}},p_{i_{k_2}},p_{i_{k_2+1}},\ldots,p_{i_{j_2-1}}\}) \cup \{p_{\bar{i}_q}\} \ . \ \square$$

IV.3.4 - Expériences numériques

Les figures 11 à 16 représentent les enveloppes convexes associées aux problèmes 1, 2, 4 - et donc 5 - de l'annexe ; le tableau suivant répertorie pour chacun des algorithmes S_2 et T_2, les points successivement introduits et éliminés au cours d'itérations ultérieures :

Problème	$\lvert H_{1,m}\rvert$	Figure	Algorithme S_2	Algorithme T_2
1	6	11 12	P_3, P_5	P_3, P_5, P_9
2	7	13 14	$P_8, P_9, P_{15}, P_{16}, P_{17}, P_{18}, P_{22}$	$P_8, P_9, P_{16}, P_{17}, P_{18}, P_{22}, P_{26}, P_{29}$
4,5	4	15 16	P_2, P_3, P_7	P_2, P_3, P_5, P_7, P_8

Le déroulement de l'algorithme S_2 pour la construction de $\bar{H}_{1,30}$ du problème 2 (figure 13) est le suivant :

$\bar{H}^*_{1,8} = \{P_1, P_8, P_{30}\}$

$\bar{H}^*_{1,9} = \{P_1, P_8, P_9, P_{30}\}$ (règle 2 : $\theta_{1,8} > \theta_{1,9}$ et $\theta_{8,30} > \theta_{9,30}$)

$\bar{H}^*_{1,10} = \bar{H}^*_{1,13} = \bar{H}^*_{1,14} = \bar{H}^*_{1,9}$ (règle 1 : $\theta_{8,9} > \theta_{8,k}$ et $\theta_{9,30} < \theta_{k,30}$ $k = 10, 13, 14$)

$\bar{H}^*_{1,15} = \{P_1, P_8, P_9, P_{15}, P_{30}\}$ (règle 2 : $\theta_{8,9} > \theta_{8,15}$ et $\theta_{9,30} > \theta_{15,30}$)

$\bar{H}^*_{1,16} = \{P_1, P_8, P_9, P_{16}, P_{30}\}$ (règle 3 : $\theta_{9,15} < \theta_{9,16}$ et $\theta_{8,9} > \theta_{8,16}$)

$\bar{H}^*_{1,17} = \{P_1, P_{17}, P_{30}\}$ (règle 3 : $\theta_{9,16} < \theta_{9,17}$; $\theta_{8,9} < \theta_{8,17}$; $\theta_{1,8} < \theta_{1,17}$)

$\bar{H}^*_{1,18} = \{P_1, P_{18}, P_{30}\}$ (règle 3 : $\theta_{1,17} < \theta_{1,18}$)

$\bar{H}^*_{1,19} = \{P_1, P_{19}, P_{30}\}$ (règle 3 : $\theta_{1,18} < \theta_{1,19}$)

$\Gamma_{1,20} = \bar{H}^*_{1,21} = \bar{H}^*_{1,19}$ (règle 1 : $\Theta_{1,19} > \Theta_{1,k}$ et $\Theta_{19,30} < \Theta_{k,30}$ k=20,21)

$\Gamma_{1,22} = \{p_1, p_{19}, p_{22}, p_{30}\}$ (règle 2 : $\Theta_{1,19} > \Theta_{1,22}$ et $\Theta_{19,30} > \Theta_{22,30}$)

$\Gamma_{1,23} = \bar{H}^*_{1,24} = \bar{H}^*_{1,22}$ (règle 1 : $\Theta_{19,22} > \Theta_{19,k}$ et $\Theta_{22,30} < \Theta_{k,30}$ k=23,24)

$\Gamma_{1,25} = \{p_1, p_{19}, p_{25}, p_{30}\}$ (règle 3 : $\Theta_{19,22} < \Theta_{19,25}$ et $\Theta_{1,19} > \Theta_{1,25}$)

$\Gamma_{1,26} = \bar{H}^*_{1,25}$ (règle 1 : $\Theta_{19,25} > \Theta_{19,26}$ et $\Theta_{25,30} < \Theta_{26,30}$)

$\Gamma_{1,u} = \bar{H}^*_{1,25} = \{p_1, p_{19}, p_{25}, p_{30}\}$. □

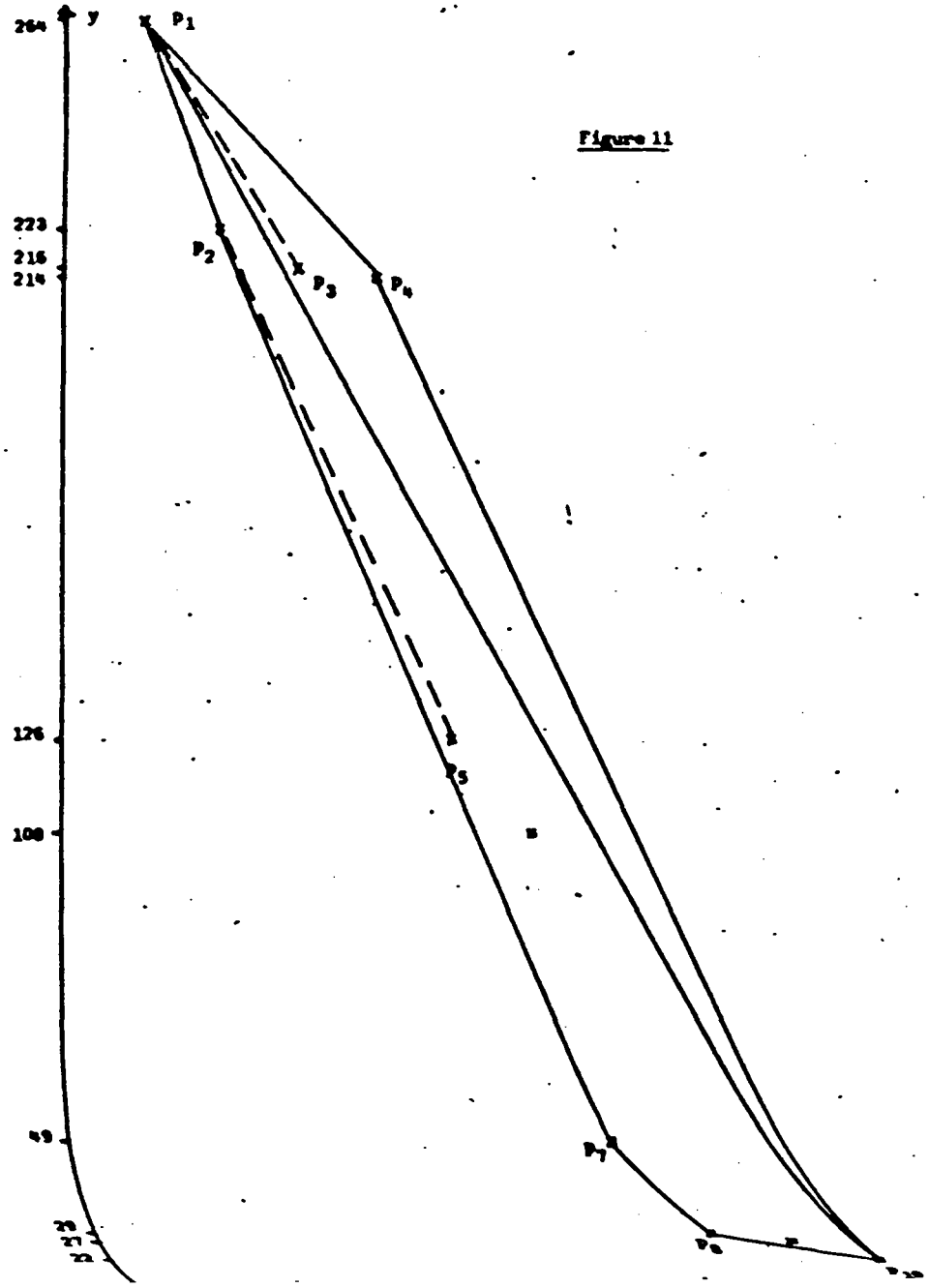


Figure 11

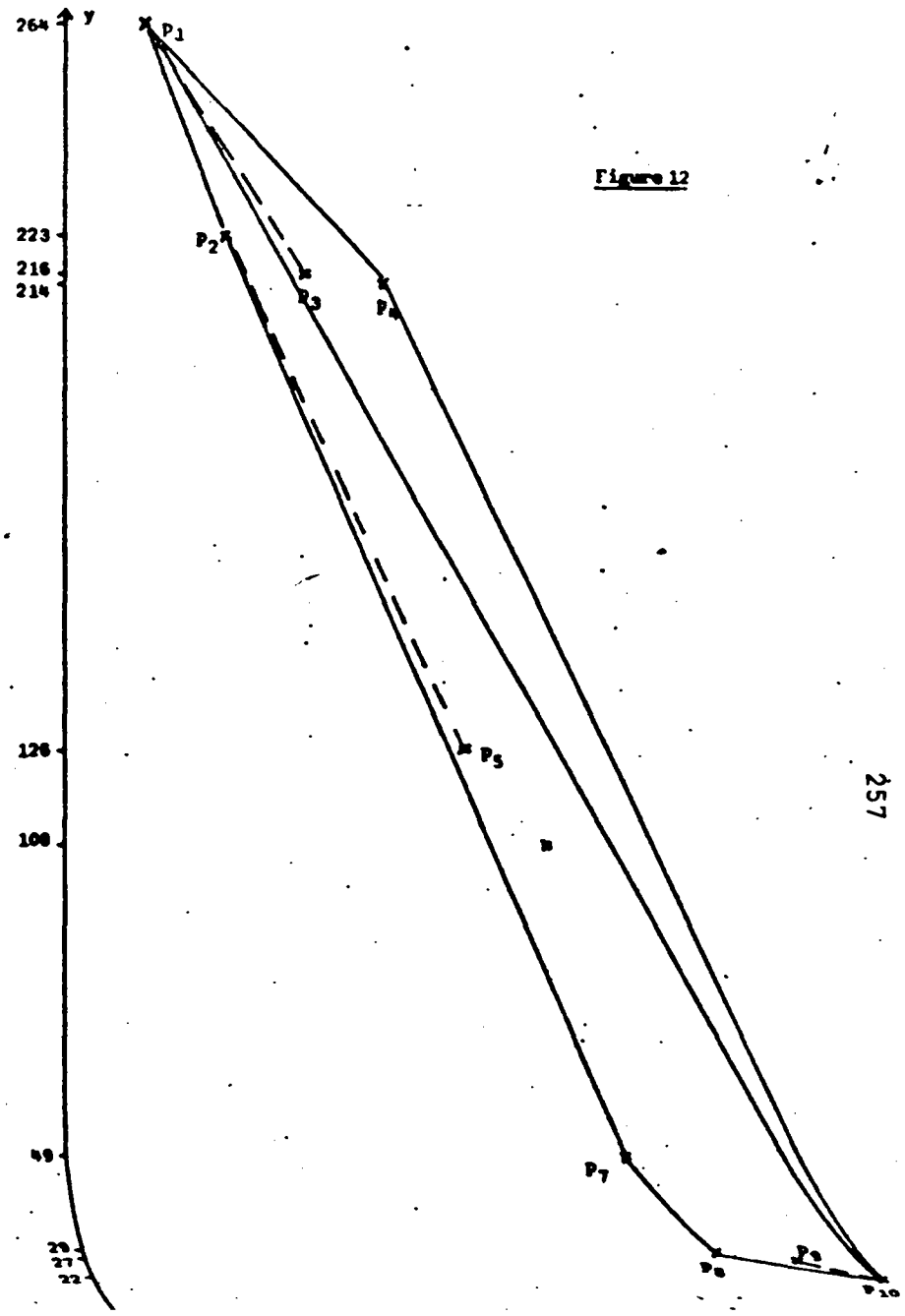


Figure 12

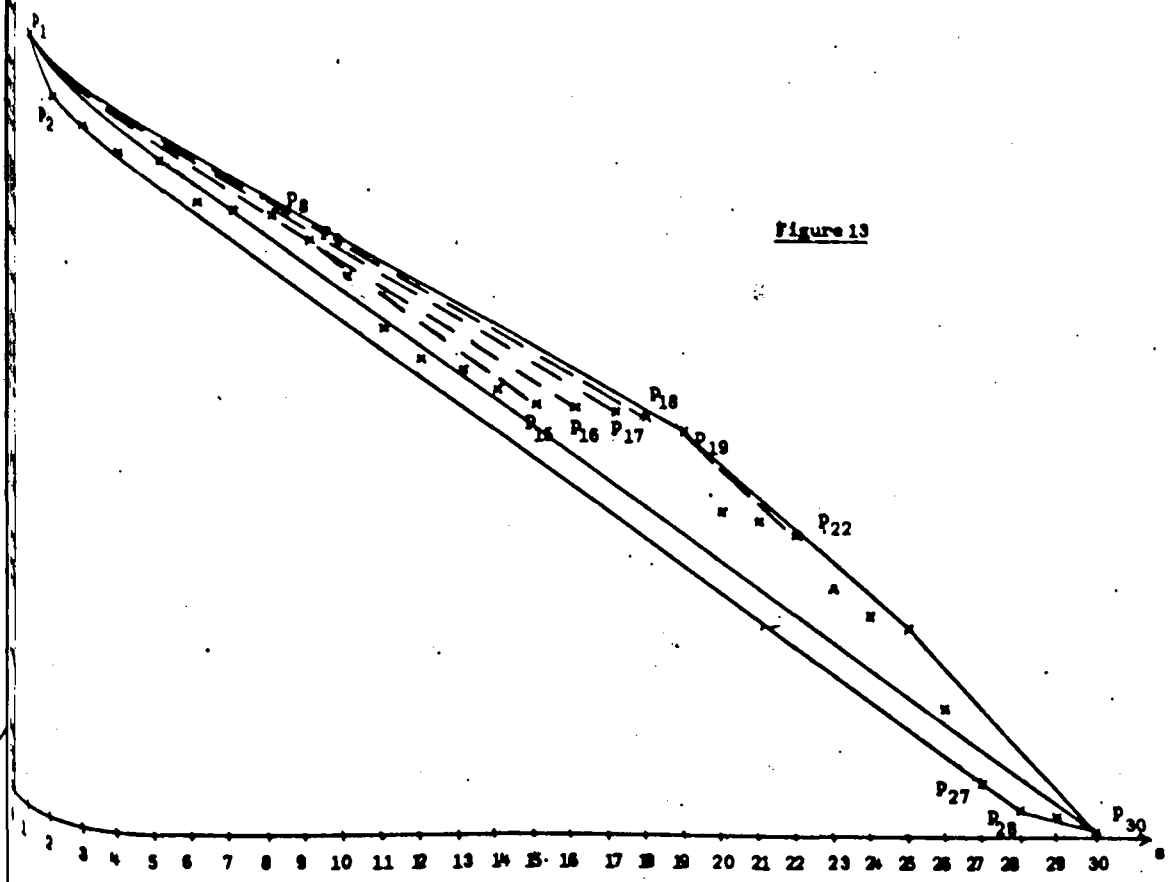


Figure 13

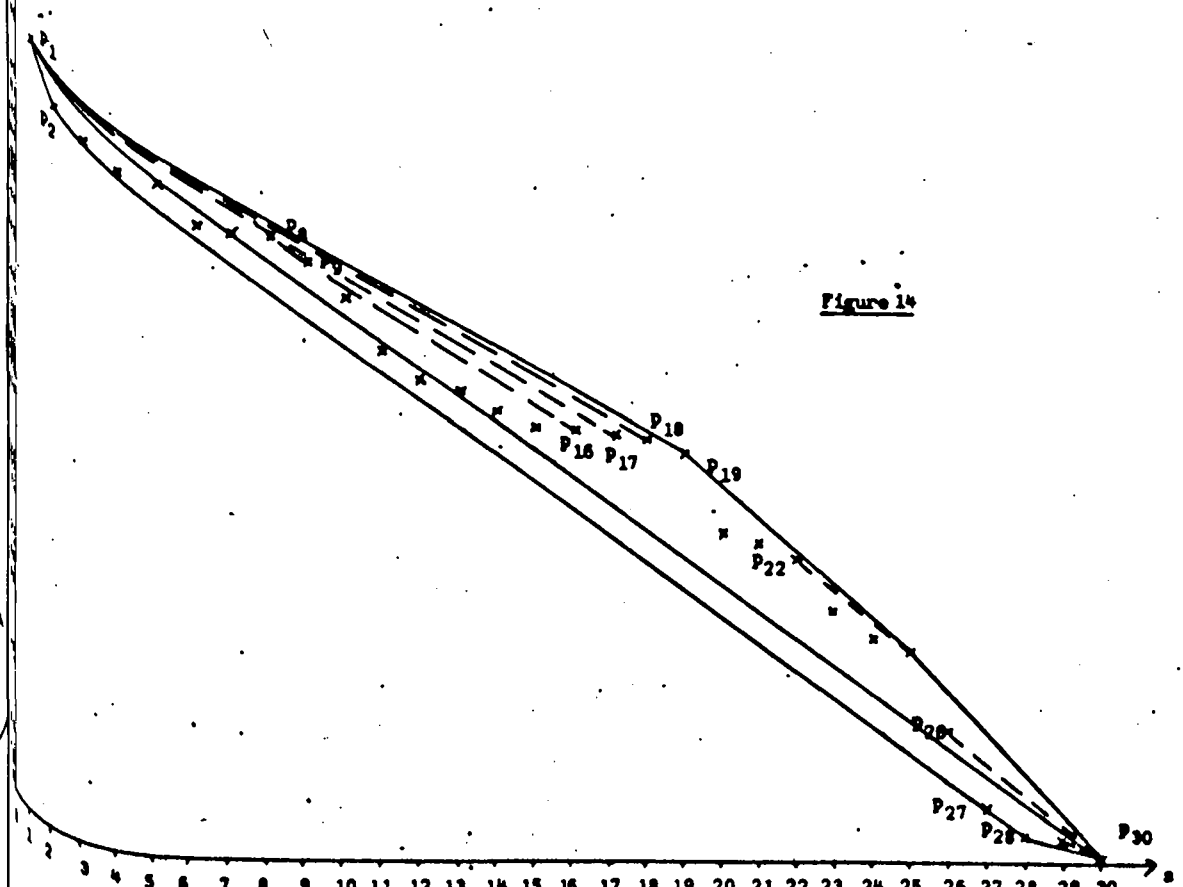


Figure 14

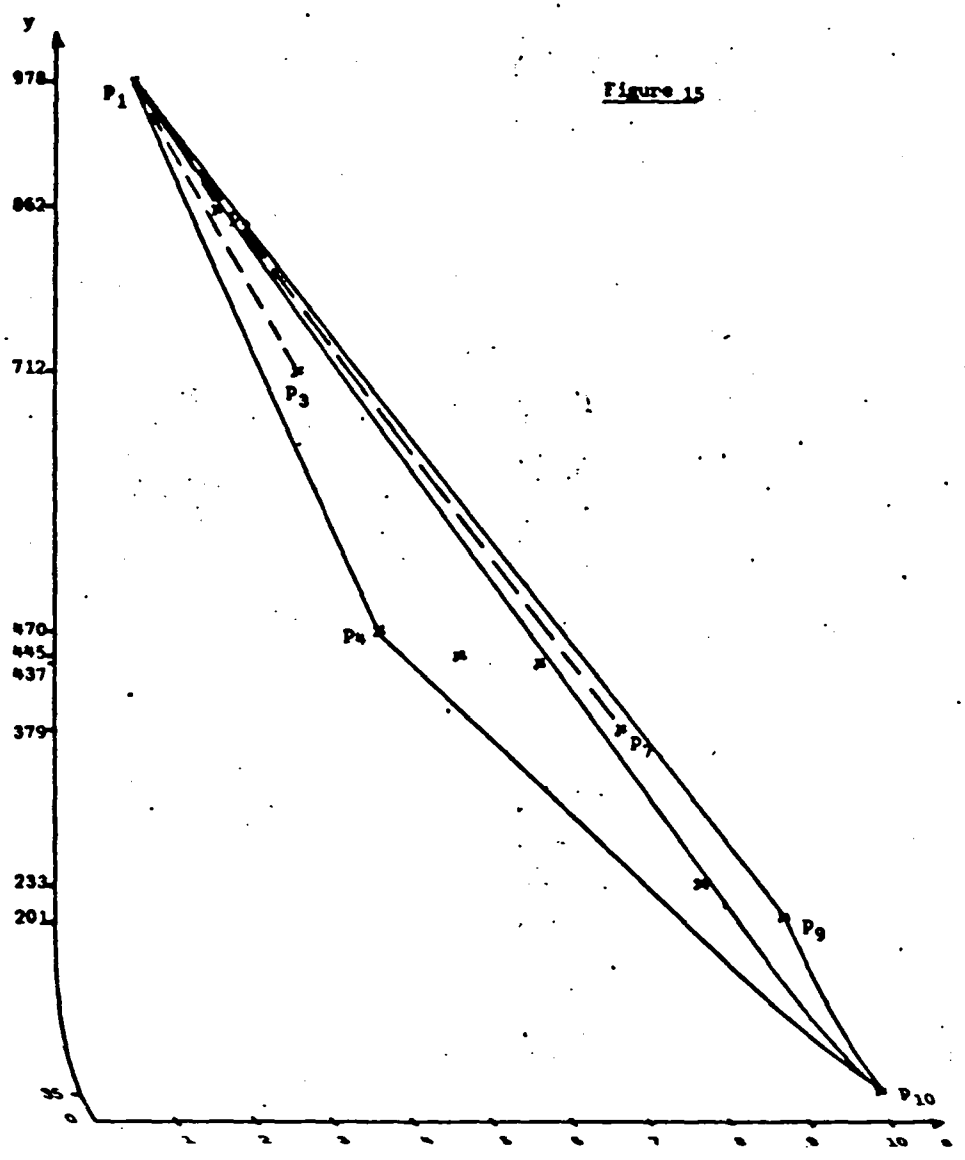


Figure 15

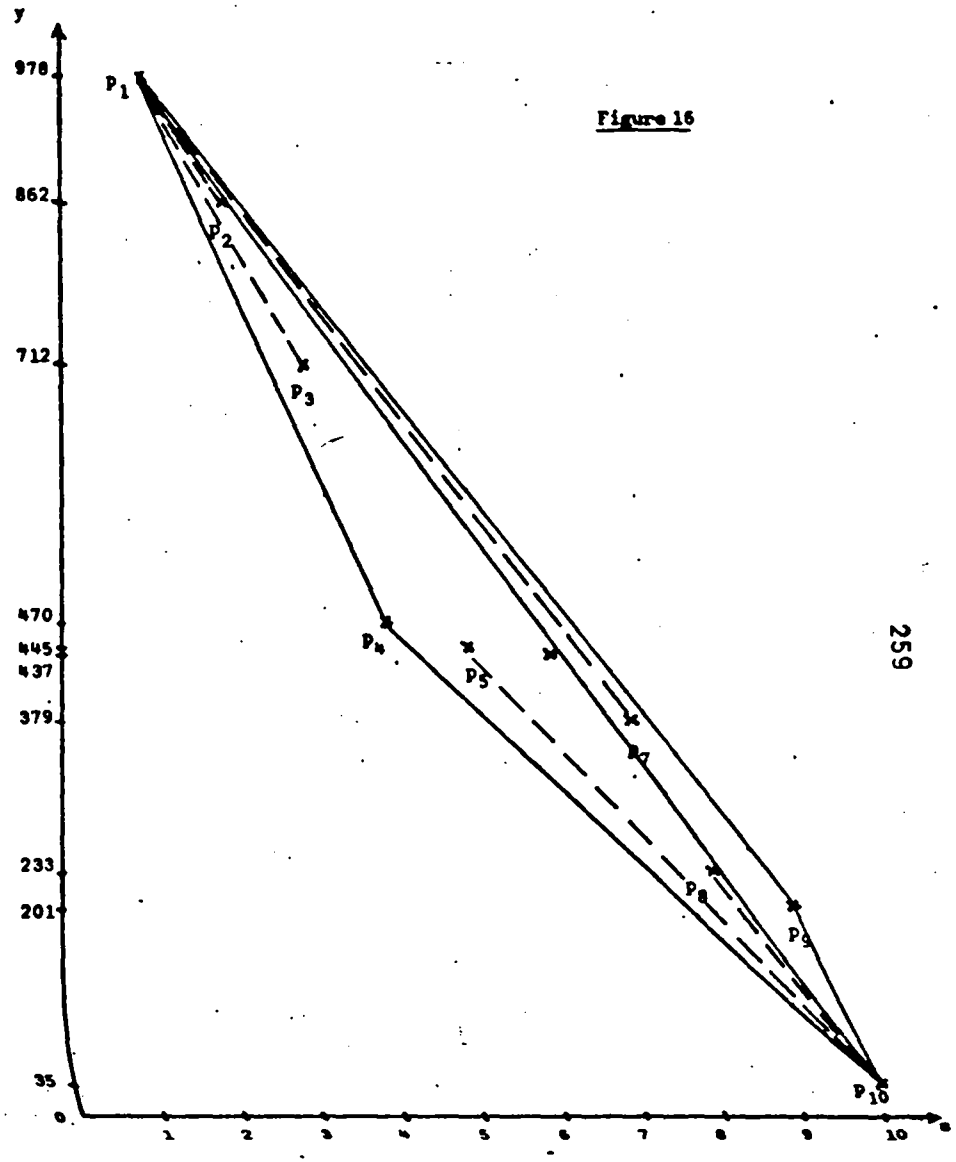


Figure 16

IV.4 - ALGORITHMES GLOUTONS POUR LES BORNES INFERIEURES DE v(K)

Deux algorithmes gloutons déterminant des bornes inférieures $\underline{v}(K) = \underline{ex}$ de $v(K)$ sont présentés dans ce paragraphe (il va de soi que les algorithmes d'encadrement de $\sum_{j=1}^{n} x_j$ fournissent également des bornes inférieures de $v(K)$ dont il faudra tenir compte dans la méthode de réduction) :

IV.4.1 - Algorithme G_1

Cet algorithme dû à Pierre Hansen, est une adaptation de l'heuristique pour le problème du knapsack classique, due à Greenberg et Hegerich [K22] :

Dans l'heuristique classique, les données sont classées définitivement suivant l'ordre décroissant des c_j/a_j, $j = 1,\ldots,n$, alors que dans l'algorithme G_1, ce classement est remis en cause après chaque fixation de variable en tenant compte de la diminution éventuelle du second membre due à la non croissance de la fonction b.

ALGORITHME G_1

0 $k \leftarrow 0$; $X_1 \equiv \emptyset$; $X_2 \equiv \{1,\ldots,n\}$; $b(0) \geq b(1)$; $q \leftarrow q^* \leftarrow \underline{v}(K) \leftarrow 0$

1 $k \leftarrow k+1$
$\underline{si}$ $k = m+1$ $\underline{alors}$ $\underline{x}_j \leftarrow 0 \quad \forall\, j \in X_2$; stop

2 $\underline{si}$ $q^* = q$ $\underline{alors}$ $q \leftarrow \min_{j \in X_2} a_j$

3 $\underline{si}$ $\sum_{j \in X_1} a_j + q > b(k)$ $\underline{alors}$ $\underline{x}_j \leftarrow 0 \quad \forall\, j \in X_2$; stop

4 $j^* : c_{j^*}/(a_{j^*}+b(k-1)-b(k)) = \max_{j \in X_2} c_j/(a_j+b(k-1)-b(k))$
$\underline{si}$ $\sum_{j \in X_1} a_j + a_{j^*} \leq b(k)$ $\underline{alors}$ $\underline{x}_{j^*} \leftarrow 1$; $\underline{v}(K) \leftarrow \underline{v}(K)+a_{j^*}$
$X_1 \equiv X_1 \cup \{j^*\}$; $X_2 \equiv X_2 \setminus \{j^*\}$
$q^* \leftarrow a_{j^*}$; $\underline{aller\ en}$ 1
$\underline{sinon}$ $\underline{x}_{j^*} \leftarrow 0$; $X_2 \equiv X_2 \setminus \{j^*\}$
$\underline{aller\ en}$ 4

Notes :

(i) La valeur de b(0) devant être choisie non inférieure à b(1), peut être en particulier l'ordonnée du point d'intersection de la droite $(p_1\ p_2)$ avec l'axe Oy, i.e. 2b(1) - b(2).

(ii) Si $\exists\ \ell : \ell \leq \sum_{j=1}^{n} x_j$ alors b(0), b(1),...,b(ℓ-1) doivent être non inférieures à b(ℓ) et telles que b(k) ≥ b(k+1) k = 0,...,ℓ-1 ; en particulier, ces valeurs peuvent être les ordonnées des points d'une droite d'appui au convexe $[M_{\ell,m}]$, passant par p_ℓ, d'abscisses respectives 0,1,...,ℓ-1. □

Théorème 41

En désignant par t ∈ {1,...,m} *le nombre de variables mises à 1, l'algorithme* G_1 *est de complexité*

(i) $O((t+1)n - t^2)$ *dans le meilleur des cas* ;

(ii) $O(n^2+t^2)$ *dans le plus mauvais cas*.

Démonstration :

(i) ∀ X_2 le nombre d'opérations pour obtenir l'indice j^* associé est borné par $k|X_2|$ où k ∈ ℕ (calcul du maximum d'une liste de $|X_2|$ nombres) ; en supposant le nombre de variables mises à 1 par l'algorithme G_1 égal à t ∈ {1,...,m}, le meilleur cas correspond à

" Pour tout X_1, X_2, k, j^* considérés $\sum_{j \in X_1} a_j + a_{j^*} \leq b(k)$ ".

Sous cette hypothèse, le nombre total d'opérations est alors borné par $k(n + (n-1) + \ldots + (n-t+1)) = k\left(tn - \frac{t(t-1)}{2}\right)$ augmenté de kn, borne associée au calcul unique de $\min_{j=1,\ldots,n} a_j$ en étape 2.

(ii) Le plus mauvais cas pour l'algorithme G_1 correspond à un problème dont les données sont telles que :

$\exists\, i_1, i_2, \ldots, i_t : a_{i_1} = \min_{j=1,\ldots,n} a_j \le a_{i_2} \le \ldots \le a_{i_t}$

$$a_{i_1} \le b(1) \text{ et } a_{i_k} \le b(k) - \sum_{\ell=1}^{k-1} a_{i_\ell} \qquad k = 2,\ldots,t$$

$\forall\, k \in \{1,\ldots,t-1\}\ c_{i_k} / (a_{i_k} + b(k-1) - b(k)) > c_{i_\ell} / (a_{i_\ell} + b(k-1) - b(k))\ \ell = k+1,\ldots,t$

$\forall\, j \neq i_k\ k \in \{1,\ldots,t\} : a_j > b(1)$ et $c_j/(a_j + b(0) - b(1)) > c_{i_1} / (a_{i_1} + b(0) - b(1))$

En effet, pour un tel problème

X_2 le test de l'étape 2 sera toujours vérifié ⟹ calcul d'un minimum d'une liste de $|X_2|$ nombres.

$X_1 \forall X_2$ le test de l'étape 3 ne sera jamais vérifié

L'étape 4 débutera par n-t calculs de maximums de listes de nombres de cardinalités successives n, n-1,...,t+1 pour fixer x_j à 0 $\forall\, j \neq i_1, i_2, \ldots, i_t$; les variables d'indices $i_1, i_2, \ldots, i_t$ seront ensuite fixées à 1, dans cet ordre avec à chaque fois le calcul du maximum d'une liste de nombres de cardinalité t, puis (t-1),..., puis 1.

En conséquence le nombre total d'opérations sera borné par

$n + (n-1) + \ldots + (t+1) + 2(t + \ldots + 1)) = k\left[\frac{n(n+1)}{2} + \frac{t(t+1)}{2}\right]$

augmenté de kn, borne associée au calcul initial de $\min_{j=1,\ldots,n} a_j$ en étape 2. □

Une variante de cet algorithme, de complexité améliorée aussi bien dans le meilleur des cas lorsque n ≥ 30 et dans le plus mauvais cas lorsque t est petit devant n($t \le \frac{n}{3}\ \forall\ n$) - ce qui sera toujours le cas dans les exemples traités (voir partie V) - fait intervenir une étape d'élimination de variables :

ALGORITHME G_1'

0 $k \leftarrow 0$; $X_1 \equiv \emptyset$; $X_2 \equiv \{1,..,n\}$; $b(0) \geq b(1)$; $q \leftarrow q^* > b(1)$; $\underline{v}(K) \leftarrow 0$

1 $k \leftarrow k + 1$

si $k = m+1$ alors $\underline{x}_j \leftarrow 0 \ \forall\, j \in X_2$; stop

2 si $q = q^*$ alors $q \leftarrow \max_{j \in X_2} a_j$

3 si $\sum_{j \in X_1} a_j + q > b(k)$ alors $E \equiv \{j \in X_2 \mid a_j > b(k) - \sum_{j \in X_1} a_j\}$

$\underline{x}_j \leftarrow 0 \ \forall\, j \in E$; si $E = X_2$ alors stop ;

$X_2 \equiv X_2 \setminus E$; $q \leftarrow \max_{j \in X_2} a_j$

4 j^* : $c_{j^*}/(a_{j^*}+b(k-1)-(b(k)) = \max_{j \in X_2} c_j/(a_j+b(k-1)-b(k))$

$x_{j^*} \leftarrow 1$; $\underline{v}(K) \leftarrow \underline{v}(K) + a_{j^*}$; $X_1 \equiv X_1 \cup \{j^*\}$; $X_2 \equiv X_2 \setminus \{j^*\}$

$q^* \leftarrow a_{j^*}$; aller en 1

Théorème 42

En désignant par $t \in \{1,\ldots,m\}$, le nombre de variables mises à 1, l'algorithme G_1' est de complexité

(i) $O(n+t^2)$ *dans le meilleur des cas*

(ii) $O(t(n-t))$ *dans le plus mauvais cas.*

Démonstration :

(i) Il est à noter que le plus mauvais cas pour l'algorithme G_1 est le meilleur pour l'algorithme G_1' :

En effet, pour le problème énoncé dans la deuxième partie de la démonstration du théorème 41 les n-t variables d'indices $j \neq i_1, i_2, \ldots, i_t$ vont être éliminées en première itération, en étape 3 ; d'où un nombre d'opérations borné par 2kn, $k \in N$ donné (il faut tenir compte du calcul initial de $\max_{j=1,\ldots,n} a_j$

Il reste alors à attribuer la valeur 1 aux t variables non éliminées, ce qui correspond à un nombre d'opérations borné par $k(t+(t-1)+\ldots+1) = k\,\frac{t(t+1)}{2}$.

Le nombre d'itérations étant égal à t, le plus mauvais cas sera tel que la $t^{ème}$ itération (mise à 1 de la $t^{ème}$ variable) soit effectuée avec un ensemble X_2 tel que $|X_2| = n-t+1$ (cardinalité maximale) ; ainsi quelle que soit l'itération $i \in \{1,\ldots,t\}$, les ensembles X_2 considérés aux étapes 2, 3 et 4 sont de cardinalité supérieure à n-t+1.

Sous cette hypothèse, le nombre global d'opérations, pour les t itérations, des étapes 3 et 4, et celui de l'étape 2 sont tous deux bornés par $k(n + (n-1) + \ldots + (n-t+1))$, d'où la conclusion. □

IV.4.2 - Algorithme G_2

Le principe de cet algorithme consiste à appliquer une séquence d'algorithmes gloutons sur des problèmes du knapsack classiques, linéarisations minorantes construites à partir de convexes emboîtés contenus dans $[M_{\ell,u}]$:

Principe général :

0 ℓ,u donnés : $\ell \le \sum_j x_j \le u$; $\underline{\ell} \leftarrow \ell$
w = nombre de variables fixées à 1 par l'algorithme G_2

1 Construction de $H_{\underline{\ell},u}$

2 Détermination d'une séparation de $[M_{\underline{\ell},u}]$ et de l'origine

3 Algorithme glouton sur le knapsack, linéarisation minorante associée
<u>si</u> w < u <u>alors</u> mise à jour de <u>v</u>(K) ; changement de coordonnées ;
$\underline{\ell} \leftarrow$ w ; <u>aller en</u> 1

4 Stop

L'exposé de l'algorithme détaillé est suivi d'une discussion sur le choix de la linéarisation (point 2 du principe général) et de l'étude de la complexité.

IV.4.2.1 - ALGORITHME DETAILLE

0 $\ell,u : \ell \leq \sum_{j=1}^{n} x_j \leq u$; $w \leftarrow 0$; $X_2 \equiv \{1,\ldots,n\}$
$\underline{u} \leftarrow u$; $\underline{\ell} \leftarrow \ell$; $\underline{v}(K) \leftarrow 0$

1 si $\ell = u$ alors $\alpha \leftarrow 0$; $\beta \leftarrow b(u)$; $X \equiv X_2$; aller en 8

2 Construction de $H_{\underline{\ell},\underline{u}}$, séquence des points extrêmes de
$[M_{\underline{\ell},\underline{u}}] = [\{p_{\underline{\ell}}, p_{\underline{\ell}+1}, \ldots, p_{\underline{u}}\}]$

3 Détermination de $y = \underline{\alpha}s + \underline{\beta}$, séparation de $[M_{\underline{\ell},\underline{u}}]$ et de l'origine

4 Elimination temporaire des variables d'indices k : $a_k + |\underline{\alpha}| > \underline{\beta}$
$X_3 \equiv X_2 \setminus \{k \mid a_k + |\underline{\alpha}| > \underline{\beta}\}$

5 si $X_3 = \emptyset$ alors aller en 12

6 si $|X_3| = 1$ (i.e. $X_3 = \{j^*\}$) alors $\underline{v}(K) \leftarrow \underline{v}(K) + a_{j^*}$;
$x_{j^*} \leftarrow 1$; $w \leftarrow w + 1$;
si $w < u$ alors aller en 11
sinon stop

7 $\alpha \leftarrow |\underline{\alpha}|$, $\beta \leftarrow \underline{\beta}$, $X \equiv X_3$

8 Résolution par l'algorithme NKR de la relaxation de
(ILK (α,β)) $\max \sum_{j\in X} c_j x_j$ sous $\sum_{j\in X} (a_j+\alpha)x_j \leq \beta$; $x_j = 0$ ou 1 $\forall\, j \in X$
(linéarisation minorante de (K) associée à la séparation $y = \alpha s+\beta$).

L'algorithme NKR fournit un sous-ensemble U de X tel que

$$\sum_{j\in U} (a_j+\alpha) \leq \beta < \sum_{j\in U\cup\{i\}} (a_j+\alpha) \; ; \forall\, j\in U \; c_j/a_j \geq c_i/a_i \geq c_k/a_k \quad \forall\, k \notin U\cup\{i\}$$

9 $\underline{x}_j \leftarrow 1 \ \forall j \in U$

$\underline{v}(K) \leftarrow \underline{v}(K) + \sum_{j \in U} c_j$

$w \leftarrow w + |U|$

<u>si</u> $w = \underline{u}$ <u>alors</u> stop

10 <u>si</u> $\ell = u$ <u>alors</u> <u>aller en</u> 12

11 a $w \leftarrow w+1$; $\underline{\ell} \leftarrow 1$; $X_2 \equiv X_2 \setminus U$

$$\left.\begin{array}{l} b(s) \leftarrow b(s) - \sum_{j \in U} a_j \\ s \leftarrow s - w + 1 \end{array}\right\} \forall s \in \{w, \ldots, \underline{u}\}$$

b <u>si</u> $b(1) \leq 0$ <u>alors</u> stop ;

c $X_2 \equiv X_2 \setminus \{k \mid a_k > b(1)\}$ (élimination des variables d'indices $k : a_k > b(1)$)

<u>si</u> $X_2 = \emptyset$ <u>alors</u> stop ;

d $\underline{u} = \max \{s \mid b(s) > 0\}$

e <u>si</u> $\underline{\ell} = \underline{u}$ <u>alors</u> $c_j^* / a_j^* = \max \{c_j/a_j \mid j \in X_2\}$;

$\underline{x}_j^* \leftarrow 1$; $\underline{v}(K) \leftarrow \underline{v}(K) + a_j^*$; stop ;

f <u>aller en</u> 2

12 algorithme glouton du type Greenberg et Hegerich sur (ILK (α,β)) : déterminer $U'' \subset X_2 \setminus (U \cup \{i\})$: $|U''| \leq \underline{u} - w$

$\underline{x}_j \leftarrow 1 \ \forall j \in U''$; $\underline{v}(K) \leftarrow \underline{v}(K) + \sum_{j \in U''} c_j$; stop

<u>Remarques</u> : Quelques explications sont à donner sur certains points particuliers de l'algorithme G_2 :

1- Lorsque $\ell = u$, cet algorithme se réduit à l'algorithme glouton classique du type Greenberg et Hegerich (test en étape 1 puis heuristique en étapes 8, 9, 10 et 12).

2- L'élimination temporaire de l'étape 4 se déduit de l'incompatibilité

des relations : $\sum_{j \in X_2} (a_j + |\underline{\alpha}|)\, x_j \leq \underline{\beta}$ et $x_k = 1$

3- Lorsqu'aucune variable n'est utilisable avec la séparation sélectionnée (étape 5), l'étape 12 achève l'algorithme glouton du type Greenberg et Hegerich sur le problème (ILK(α,β)) dont les paramètres sont ceux de la séparation de l'itération précédente.

4- L'étape 6 traite le cas particulier où une seule variable est utilisable temporairement (i.e. dans le problème linéarisé (ILK ($|\underline{\alpha}|$, $\underline{\beta}$))).

5- Les étapes 8 et 9 concernent l'algorithme glouton classique appliqué à (ILK(α,β)) avec un test d'arrêt en étape 9. Il est à noter que
$\forall s \; \forall x$: $ax \leq \underline{\alpha} s + \underline{\beta} \Rightarrow ax \leq b(s)$, d'où $a\underline{x} \leq b(w+|U|)$.

6- La mise à jour des données nécessaire à l'exécution de l'itération suivante est faite en étape 11 :

. <u>étape 11-a</u> : suppression des points $p_w, \ldots, p_{w+|U|}$ (par la mise à jour de $\underline{\ell}$), changement de coordonnées des points $p_{\underline{\ell}}, \ldots, p_u$ (b(s) diminué de $\sum_{j \in U} a_j$ s = $\underline{\ell}, \ldots, \underline{u}$; nouvelles abscisses respectives $1, \ldots, \underline{u}-\underline{\ell}+1$)

. <u>étape 11-b</u> : $0 \geq b(1) \geq b(2) \geq \ldots \geq b(\underline{u}-\underline{\ell}+1) \Rightarrow$ aucune variable ne peut plus être utilisée.

. <u>étape 11-c</u> : après élimination des variables d'indices k : $a_k > b(1)$, l'algorithme s'achève si aucune des variables n'est utilisable.

. <u>étape 11-d</u> : mise à jour de la borne supérieure $\underline{u}$ de Σx_j, qui tient compte à la fois du changement d'abscisse et de la nécessité de n'envisager que les seconds membres positifs.

. <u>étape 11-e</u> : si une seule variable est à mettre à 1, l'algorithme glouton se réduit à la recherche du rapport maximum des c_j/a_j, $j \in X_2$ puisque, d'après l'étape 11-c, $a_j \leq b(1)$ $\forall$ $j \in X_2$.

. étape 11.f : une nouvelle itération peut être mise en jeu. □

IV.4.2.2 - CHOIX DE LA LINEARISATION*

a) Principe :

Etant donné $\underline{H}_{\ell,u} = \{p_\ell, p_{h_1}, \ldots, p_{h_r}, p_u\}$, parmi les nombreux choix possibles de séparation de $[M_{\ell,u}]$ et de l'origine, on citera en particulier :

(i) $(p_\ell\ p_{h_1})$ ou $(p_{h_1}\ p_{h_2}), \ldots,$ ou $(p_{h_r}\ p_u)$ droites d'appui à $[M_{\ell,u}]$ telles que

$$(p_k\ p_j) \cap [M_{\ell,u}] = [p_k\ p_j] \qquad (k,j) = (\ell,h_1),(h_1,h_2),\ldots,(h_r,u)$$

(ii) D_ℓ, ou $D_{h_1}, \ldots,$ ou D_u droites d'appui à $[M_{\ell,u}]$ telles que

a) $D_k \cap [M_{\ell,u}] = \{p_k\} \qquad k = \ell, h_1, \ldots, h_r, u$

b) en notant α_k la pente de D_k $\forall\ k \in \{\ell,h_1,\ldots,h_r,u\}$,

$$-\infty < \alpha_\ell < \theta_{\ell,h_1} < \alpha_{h_1} < \theta_{h_1,h_2} < \alpha_{h_2} < \ldots < \alpha_{h_r} < \theta_{h_r,u} < \alpha_u \leq 0$$

Note : Dans le cas particulier où $\underline{H}_{\ell,u} = \{p_\ell, p_u\}$, ces choix se réduisent à $(p_\ell\ p_u)$ ou D_ℓ ou D_u avec $-\infty < \alpha_\ell < \theta_{\ell,u} < \alpha_u \leq 0$. □

Dans le but,

- d'une part, de choisir une linéarisation qui "épouse" au mieux la forme du convexe $[M_{\ell,u}]$ (dans le sens de son allongement)

- d'autre part, de minimiser le volume des calculs

nous avons pris le parti de procéder de la manière suivante :

---

* Toute la discussion de cette section peut être transposée pour la linéarisation majorante en raisonnant à partir de $\bar{H}_{\ell,u}$ au lieu de $\underline{H}_{\ell,u}$.

- Si $\underline{H}_{\ell,u} = \{p_\ell, p_u\}$, on considère $(p_\ell\ p_u)$ d'équation

$$\tilde{y} = b(u) + \theta_{\ell,u}\ (s-u)$$

- Si $\underline{H}_{\ell,u} \supsetneq \{p_\ell, p_u\}$, on considère D_{h^*}, parallèle à $(p_\ell\ p_u)$,

avec h^* tel que $d(p_{h^*},(p_\ell p_u)) = \max_{p_k \in \underline{H}_{\ell,u}} d(p_k,(p_\ell p_u))$

dont l'équation est

$$\tilde{y} = b(h^*) + \theta_{\ell,u}(s-h^*).$$

La séparation choisie en définitive est la droite d'équation

$$y = \underline{\alpha}s + \underline{\beta} \text{ telle que } \begin{cases} \underline{\alpha} = \lfloor \theta_{\ell,u} \rfloor \\ \underline{\beta} = \lfloor b(k) - k\theta_{\ell,u} \rfloor \text{ avec } k = \begin{cases} u & \text{si } \underline{H}_{\ell,u} = \{p_\ell, p_u\} \\ h^* & \text{sinon} \end{cases} \end{cases}$$

à laquelle sera associé un problème linéarisé à coefficients tous entiers à l'instar du problème d'origine ; de plus, les choix de $\underline{\alpha}$ et de $\underline{\beta}$ impliquent que y sépare $\tilde{y}$ de l'origine et donc répond à la question.

Note : les choix de $\underline{\alpha}$ et de $\underline{\beta}$ ne sont bien évidemment pas à tous coups les meilleurs, cependant ils permettent d'aboutir par l'algorithme G_2 aux valeurs optimales des quatre premiers problèmes proposés par Guignard, Posner [EK9].

b) Complexité :

Une fois adopté le principe du choix de la séparation, il reste à définir la méthode la moins coûteuse pour déterminer p_{h^*}. En posant $t = |\underline{H}_{\ell,u}|$, il est clair que toutes les méthodes seront de complexité $O(t)$

recherche du maximum d'une liste de nombres) ; elles ne différeront donc les unes des autres que par la constante multiplicative de t. Deux types de méthodes sont présentées :

- Méthodes 1 et 2 : elles utilisent chacune la distance euclidienne de $p_k \in \underline{H}_{\ell,u}$ à $(p_\ell p_u)$, les nombres d'opérations pour chaque p_k étant respectivement 11 (y compris 2 appels de fonctions) et 13, les méthodes 1 et 2 sont d'efficacités comparables.

- Méthode 3 : utilisant un multiple (indépendant de p_k) de la distance euclidienne de chaque point de $\underline{H}_{\ell,u}$ à $(p_\ell p_u)$, elle s'avère au moins quatre fois plus rapide que les méthodes 1 et 2 qui sont donc données pour information.

Note : $p_h{}^*$ peut être déterminé en calculant directement $\max_{p_k \in \underline{M}_{\ell,u}} d(p_k,(p_\ell p_u))$; en remarquant que la méthode 3 est plus rapide que la construction de $[\underline{M}_{\ell,u}]$, il est clair qu'avec le choix de $\underline{\beta}$ énoncé au a), la détermination de $\underline{H}_{\ell,u}$ préalable à celle $p_h{}^*$ est moins efficace que le calcul direct - ce qui n'est pas le cas pour les méthodes 1 et 2 - le code comportera néanmoins la séquence des constructions de $\underline{H}_{\ell,u}$ et de $p_h{}^*$ laissant ainsi la possibilité de choisir un autre type de séparation. □

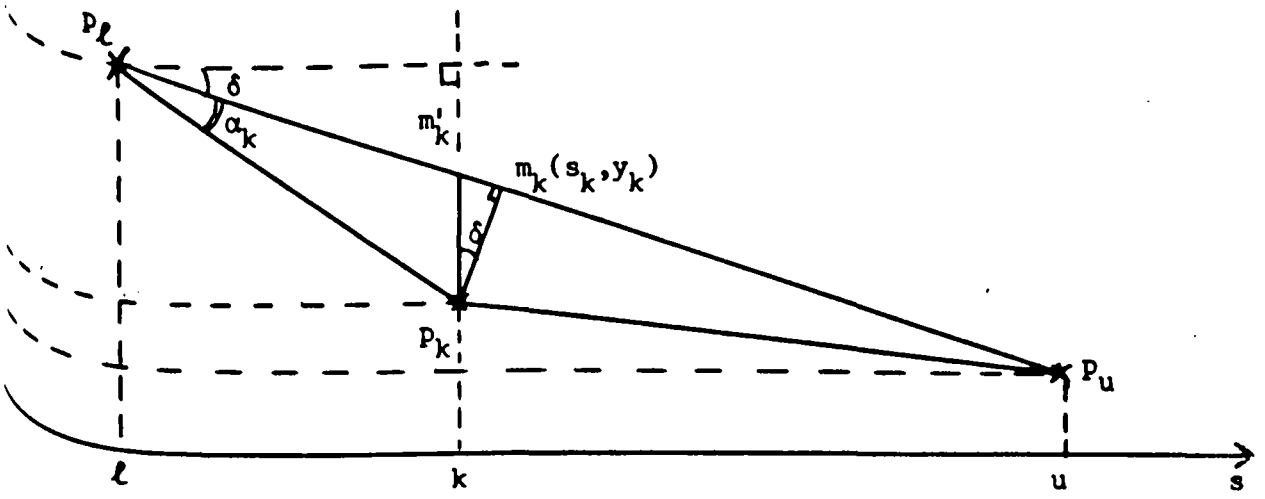


Figure 17

En supposant connue l'équation de $(p_\ell p_u)$:

$$y_1 = \theta_{\ell,u}\, s + \gamma \quad \text{avec } \gamma = b(u) - u\,\theta_{\ell,u}$$

les trois méthodes sont les suivantes (se référer à la figure 17) :

(i) Méthode 1 :

En posant :

- m_k : projection de p_k sur $(p_\ell p_u)$
- α_k : angle opposé à $[p_k\ m_k]$ dans le triangle $p_k p_\ell m_k$

$$\forall\ p_k \in \underline{H}_{\ell,u} \quad d(p_k,(p_\ell p_u)) = d(p_k,m_k) = d(p_\ell,p_k) * \sin \alpha_k$$

En remarquant que $\alpha_k = \text{Arctg}\ \theta_{\ell,u} - \text{Arctg}\ \theta_{\ell,k}$ et en considérant le carré de la distance pour minimiser le nombre d'appels de procédures, il faut donc calculer pour chaque $p_k \in \underline{H}_{\ell,u}$:

$$\theta_{\ell,k} = \frac{b(k) - b(\ell)}{k-\ell} \text{ puis } ((b(\ell) - b(k))^2 + (\ell-k)^2) * \sin^2(\text{Arctg}\ \theta_{\ell,u} - \text{Arctg}\ \theta_{\ell,k})$$

ce qui correspond à 11 "opérations" se décomposant en 4 additions/soustractions 5 multiplications/divisions et 2 appels de procédures (Arctg $\theta_{\ell,u}$ est calculé une seule fois en début de méthode).

(ii) Méthode 2 :

Les coordonnées (s_k,y_k) de m_k sont calculées comme suit :

$$- m_k \in (p_\ell p_u) \Longrightarrow y_k = \theta_{\ell,u}\, s_k + \gamma \qquad (77)$$

- l'équation de $(p_k\ m_k)$ orthogonale à $(p_\ell p_u)$ est de la forme $y = -s/\theta_{\ell,u} + b(k) + k/\theta_{\ell,u}$, donc en particulier

$$y_k = - s_k/\theta_{\ell,u} + b(k) + k/\theta_{\ell,u} \qquad (78)$$

$$(77) \text{ et } (78) \Longrightarrow \begin{cases} s_k = (\Theta_{\ell,u} * (b(k) - \gamma) + k)/(\Theta^2_{\ell,u} + 1) \\ y_k = \Theta_{\ell,u} * s_k + \gamma \end{cases}$$

$$\Longrightarrow d^2(p_k,m_k) = (s_k - k)^2 + (y_k - b(k))^2$$

ce qui correspond à 13 opérations se décomposant en 7 additions/soustractions et 6 multiplications/divisions.

(iii) Méthode 3 :

Puisqu'il suffit de trouver le point de $\underline{H}_{\ell,u}$ le plus éloigné de $(p_\ell p_u)$, cette méthode ne calcule pas la distance euclidienne de p_k à m_k mais celle de p_k à m'_k, point d'intersection de $[p_\ell p_u]$ avec la droite s = k, puisque

$$\forall\, p_k \in \underline{H}_{\ell,u}\ d(p_k,m_k) = d(p_k,m'_k) * \cos\delta \qquad (79)$$

où $\delta = \text{Arctg}\ \Theta_{\ell,u}$ est indépendant de p_k.

Ce calcul ne nécessite que 3 opérations puisqu'à l'évidence :

$$d(p_k,m'_k) = k * \Theta_{\ell,u} + \gamma - b(k) \qquad (80)$$

Notes :

1) $Q_k = d(p_\ell,p_k) + d(p_k,p_u) < Q_j = d(p_\ell,p_j) + d(p_j,p_u) \not\Rightarrow d(p_k,m_k) < d(p_j,m_j)$ (contre exemple : pour $d(p_\ell,p_u) = 90$; $d(p_k,m_k) = 11$ et $d(p_\ell,m_k) = 45$; $d(p_j,m_j) = 10$ et $d(p_\ell,m_j) = 10$; on aboutit à $Q_k \simeq 92.65$ et $Q_j \simeq 94.76$. □

2) Il est à remarquer que le calcul effectif de $d(p_k,m_k)$ par la méthode 3 est de surcroît plus rapide que celui des méthodes 1 et 2 : le nombre d'opérations, égal à 5 (d'après (79) et (80)), se décompose en 2 additions/soustractions, 2 multiplications et un appel de procédure. □

IV.4.2.3 - COMPLEXITE DE L'ALGORITHME G_2

Théorème 43

En désignant par $t \in \{1,\ldots,m\}$ *le nombre de variables mises à 1, l'algorithme* G_2 *est de complexité*

(i) $O(n)$ *dans le meilleur des cas*

(ii) $O(t(n-t))$ *dans le plus mauvais cas*

Démonstration :

(i) Dans le meilleur des cas, l'algorithme G_2 détermine les t variables mises à 1 dès la résolution de la première linéarisation minorante ; le nombre global d'opérations sera donc borné par

$k \in \mathbb{N}$ donné,
- kn (construction d'une enveloppe convexe - étapes 2 et 3)
- + kn (élimination temporaire de variables - étape 4)
- + kn (résolution par l'algorithme NKR du problème linéarisé et fixation des t variables à 1 - étapes 7, 8 et 9).

d'où la conclusion.

(ii) Le plus mauvais cas correspond à la fixation d'une seule variable à 1 à chaque itération ; i.e. pour un ensemble X_2 associé à une itération, le nombre d'opérations associé sera borné par :

$k \in \mathbb{N}$ donné,
- $3k\,|X_2|$ (étapes 2,3,4,7,8 et 9)
- $+k(|X_2|-1)$ (élimination définitive de variables - étape 11)

Le nombre global d'opérations correspondant aux t itérations est donc majoré par

$k([3n+n-1] + [3(n-1)+n-2] + \ldots + [3(n-t+1)+n-t]) = k(4nt - 2t^2 + t)$. □

IV.4.2.4 - EXEMPLES DE DEROULEMENT DE L'ALGORITHME G_2

(i) Problème test n°2 : n = m = 30 (voir Annexe)

a) Hypothèse de départ : $\sum_{j=1}^{n} x_j \leq 11$: - i.e. $\underline{\ell} = 1$, $\underline{u} = 11$ - (figure 18):

Itération 1 : $X_2 = \{1,2,\ldots,30\}$

$H_{1,11} = \{p_1, p_2, p_{11}, p_{10}, p_9\}$ (séquence des points extrêmes de $[M_{1,11}]$).

p_2 étant l'unique point sous $[p_1\ p_{11}]$ dont la pente est -10.2, la droite de séparation de $[M_{1,11}]$ et de l'origine est y = -11s + 289, d'où le problème linéarisé :

$$(ILK\ (11,289))\ \max \sum_{j\in X_2} c_j x_j \ \&c. \sum_{j\in X_2} (a_j+11)\, x_j \leq 289;\ x_j = 0 \text{ ou } 1 \quad \forall\, j \in X_2$$

dont toutes les variables sont utilisables (i.e. $X_3 \equiv X_2$).

L'algorithme NKR fournit U = {1,7,11,16,12,9,20,8} et i = 2 tels que

$$\sum_{j\in U} (a_j+11) = 238 < 289 < \sum_{j\in U} (a_j+11) + a_{21} + 11 = 309$$

$|U| < 11 \Longrightarrow \underline{x}_j = 1\ \forall\, j \in U$; $\underline{v}(K) = \sum_{j\in U} c_j = 451$; $X_2 \equiv X_2 \setminus U$;

Note : $\sum_{j\in U} a_j + a_{21} = 150 + 60 = 210 < b(9) = 221 \Longrightarrow$ en ayant pour référence le problème (K) la variable x_{21} aurait pu être fixée à 1. □

Itération 2 : $w = |U| = 8 \Longrightarrow \underline{\ell} = 1 \leq \sum_{j\in X_2} x_j \leq 3 = \underline{u}$

Les coordonnées de p_9, p_{10} et p_{11} deviennent respectivement (1,71), (2,57), (3,38)

$E = \{k \mid a_k > b(1) = 71\} = \{2,13,15,22,23,28,30\} \Longrightarrow X_2 \equiv X_2 \setminus E$

$H_{9,11} = \{p_9, p_{11}, p_{10}\} = M_{9,11}$, la droite de séparation de $[M_{9,11}]$ et de l'origine étant y = -17s + 87 (droite de pente entière la plus proche de celle de $(p_9 p_{11})$), le problème linéarisé (à $|X_2| = 15$ variables) est le suivant :

(ILK (17,87)) max $\sum_{j\in X_2} c_j x_j$ s.c. $\sum_{j\in X_2} (a_j+17)x_j \leq 87$; $x_j=0$ ou 1 $\forall$ $j \in X_2$

dont toutes les variables sont utilisables (i.e. $X_3 \equiv X_2$).

L'algorithme NKR fournit U = {21} et i = 26 tels que

$$a_{21} + 17 = 77 < 87 < a_{21} + a_{26} + 2 \times 17 = 159$$

$w + |U| = 9 < 11 \Longrightarrow \underline{x}_{21} = 1$ et $\underline{v}(K) = 451 + 93 = 544$; $X_2 \equiv X_2 \setminus \{21\}$

<u>itération 3</u> : $w = w + |U| = 9$; $\underline{\ell} = 1 \leq \sum_{j\in X_2} x_j \leq 2 = \underline{u}$

Les coordonnées de p_{10} et p_{11} deviennent respectivement (1,-3) et (2,-21).
$b(1) = -3 < 0 \Longrightarrow$ le sous-problème

$$\max \sum_{j\in X_2} c_j x_j \text{ s.c. } \sum_{j\in X_2} a_j x_j \leq b\Big(\sum_{j\in X_2} x_j\Big) ; 1 \leq \sum_{j\in X_2} x_j \leq 2 ; x_j=0 \text{ ou } 1 \quad \forall j \in X_2$$

ayant un domaine vide, l'algorithme G_2 s'achève :

$\underline{x}_j = 1$ j=1,7,8,9,11,12,16,20,21 et $\underline{v}(K) = 544$.

b) <u>Hypothèse de départ</u> $4 \leq \sum_{j\in X_2} x_j \leq 11$: - i.e. $\underline{\ell}=4$ et $\underline{u}=11$ - (figure 19) :

L'unique différence avec le déroulement précédent réside dans le choix de la linéarisation de l'itération 1 ; le premier convexe envisagé est $[M_{4,11}] \Longrightarrow H_{4,11} = \{4,11,10,9,8,5\}$ d'où la droite de séparation $y = -10s + 288$ qui aboutit à un problème linéarisé conduisant à des fixations de variables identiques au a).

(ii) <u>Problème test n°5</u> : n = 100, m = 10 (voir Annexe)

a) <u>Hypothèse de départ</u> $\sum_{j=1}^{n} x_j \leq 7$: - i.e. $\underline{\ell}=1, \underline{u}=7$ - (figure 20) :

<u>itération 1</u> : $X_2 = \{1,2,\ldots,100\}$

$H_{1,7} = \{p_1, p_4, p_7\}$; p_4 est l'unique point de $H_{1,7}$ n'appartenant pas à $(p_1\ p_7)$ d'où la droite de séparation $y = -100s + 869$ et le problème linéarisé :

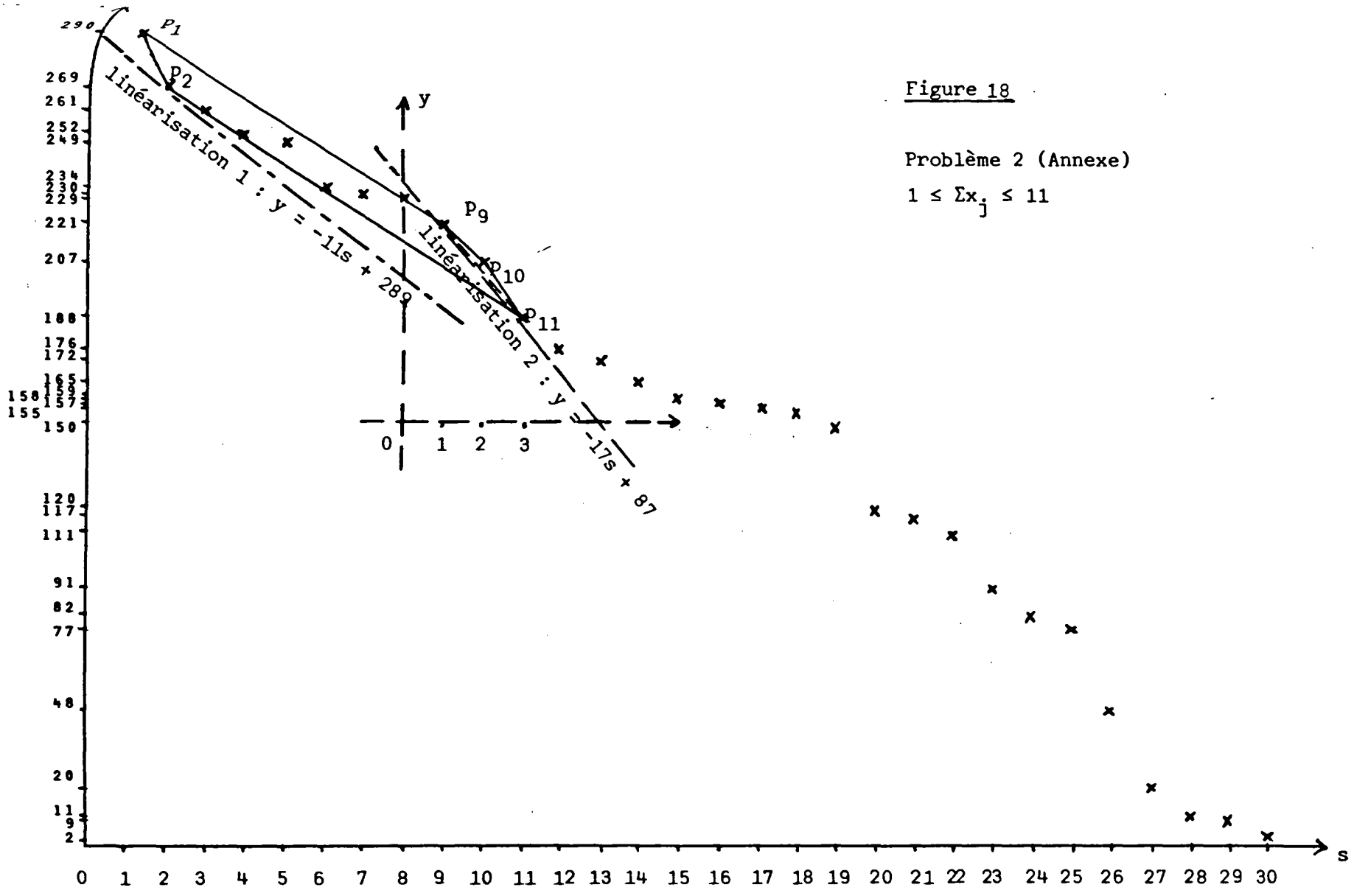


Figure 18

Problème 2 (Annexe)

$1 \leq \Sigma x_j \leq 11$

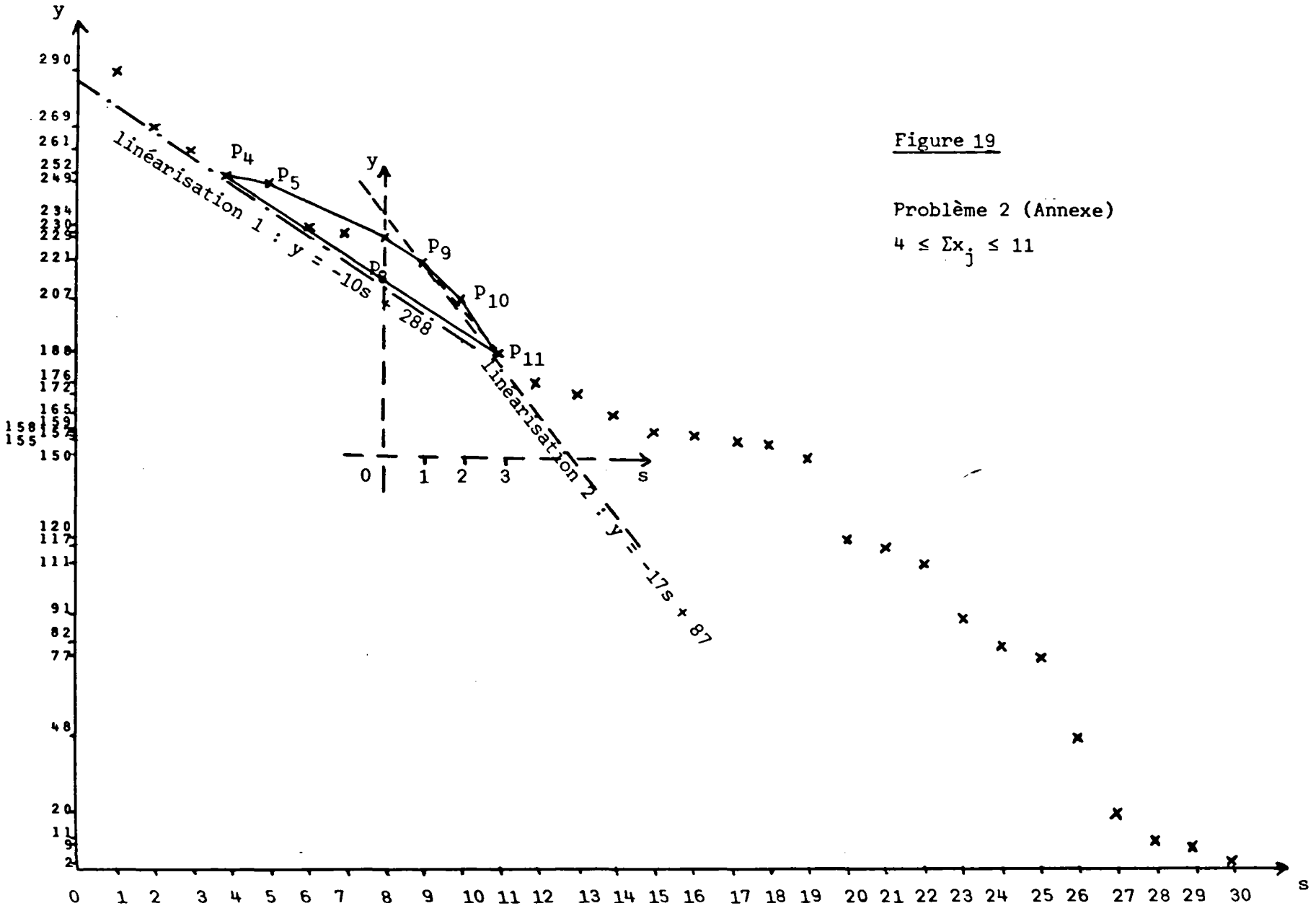


Figure 19

Problème 2 (Annexe)

$4 \leq \Sigma x_j \leq 11$

(ILK(100,869)) max $\sum_{j\in X_2} c_j x_j$ s.c. $\sum_{j\in X_2} (a_j+100)\, x_j \leq 869$; $x_j = 0$ ou 1 $\forall\, j \in X_2$

dont une variable n'est pas utilisable (la variable x_{74} est éliminée définitivement à 0 puisque $a_{74} = 1000 > b(1) = 978$) (i.e. $X_2 = X_2 \setminus \{74\}$).

L'algorithme NKR fournit $U = \{28,31,42,65\}$ et $i = \{59\}$ tels que

$$\sum_{j\in U} (a_j+100) = 676 < 869 < \sum_{j\in U} (a_j+100) + a_{59} + 100 = 926$$

$|U| = 4$: $\underline{x}_j = 1$ $\forall\, j \in U$; $\underline{v}(K) = \sum_{j\in U} c_j = 923$; $X_2 \equiv X_2 \setminus U$

Note : $\sum_{j\in U} a_j + a_{59} = 276 + 150 = 426 < b(5) = 445 \Longrightarrow$

$$\sum_{j\in U} c_j + c_{59} = 923 + 256 = 1179$$

mais alors $b(6) = 437 - 426 = 9 < a_k$ $\forall\, k \in X_2$ $\Longrightarrow$ aucune autre variable ne pourrait être mise à 1, et la valeur obtenue est inférieure à celle de l'algorithme G_2. □

itération 2 : $w = |U| = 4 \Longrightarrow \underline{\ell} = 1 \leq \sum_{j\in X_2} x_j \leq 3 = \underline{u}$

Les coordonnées de p_5, p_6 et p_7 deviennent respectivement (1,169), (2,161), (3,103), $|E| = |\{k \mid a_k > b(1)\}| = 89$ variables sont éliminées et prennent la valeur 0.

$\Longrightarrow X_2 = \{3,50,51,59,67,70,86\}$

$H_{5,7} = \{p_5,p_7,p_6\} = M_{5,7}$, la droite de séparation de $[M_{5,7}]$ et de l'origine étant $y = -33s + 202$ (droite de pente entière la plus proche de celle de (p_5p_7)), le problème linéarisé est le suivant :

$$\text{(ILK (33,202))} \quad \left[\begin{array}{l} \max\ 256x_{59}+226x_{86}+11x_{70}+151x_3+115x_{51}+4x_{50}+92x_{67} \\ \text{s.c. } 183x_{59}+158x_{86}+94x_{70}+99x_3+67x_{51}+91x_{50}+186x_{67} \leq 202 \\ \qquad x_j = 0 \text{ ou } 1 \quad \forall\, j \end{array}\right.$$

L'algorithme NKR fournit $U = \{51,3\}$ et $i = \{86\}$ tels que

$$a_{51} + a_3 + 2 \times 33 = 166 < 202 < a_{51} + a_3 + a_{86} + 3 \times 33 = 324$$

$w + |U| = 6 < 7 \Longrightarrow \underline{x}_{51} = \underline{x}_3 = 1$ et $\underline{v}(K) = 923 + 115 + 151 = 1189$

<u>itération 3</u> : $w = w + |U| = 6$; $\sum_{j \in X_2} x_j = 1 = \underline{\ell} = \underline{u}$ où $X_2 = \{50,59,67,70,86\}$

les coordonnées de p_7 deviennent $(1,3) \Longrightarrow \forall\, k \in X_2\ a_k > 3$ et l'algorithme G_2 s'achève en ayant fourni :

$\underline{v}(K) = 1189$ avec $\underline{x}_j = 1$ $j=3,28,31,42,51,65.$

b) <u>Hypothèse de départ</u> $2 \leq \Sigma x_j \leq 7$: - i.e. $\underline{\ell}=2$ et $\underline{u}=7$ - (figure 21)

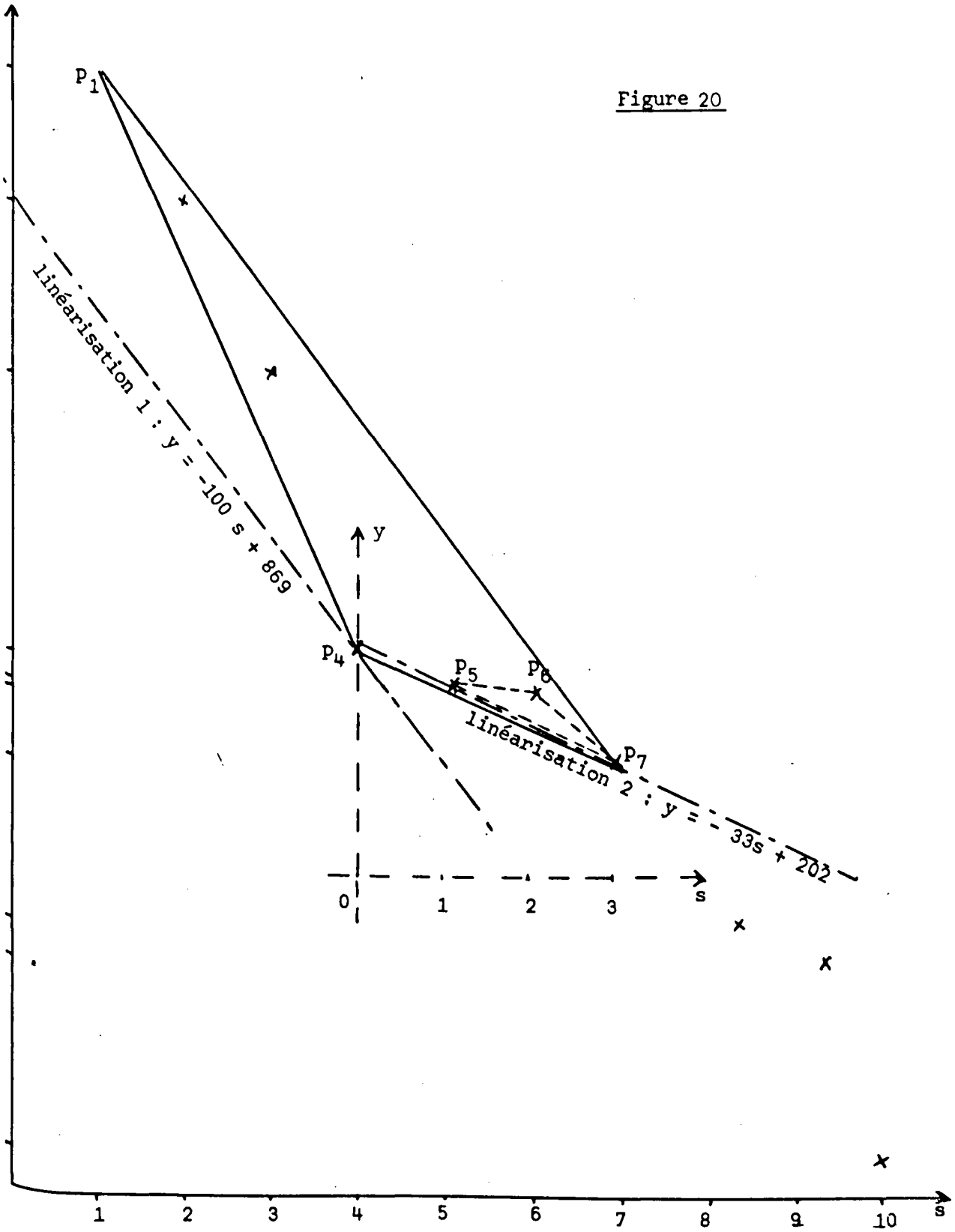
Comme pour l'exemple 2, la différence avec le déroulement du a) réside dans le choix de la linéarisation de l'itération 1 ; le premier convexe envisagé est $[M_{2,7}] \Longrightarrow H_{2,7} = \{p_2,p_4,p_7\}$ d'où la droite de séparation $y = -97s + 856$ qui aboutit à un problème linéarisé permettant de fixer à la valeur 1 les variables identiques au a) ; il faut toutefois noter que ce problème possède 77 variables (au lieu de 99 dans a)) puisque

- d'une part, $|\{k \mid a_k > b(2) = 862\}| = 9 \Longrightarrow |X_2| = 90$

- d'autre part, $|\{k \in X_2 \mid a_k + 97 > 856\}| = 13 \Longrightarrow |X_3| = 77.$ □

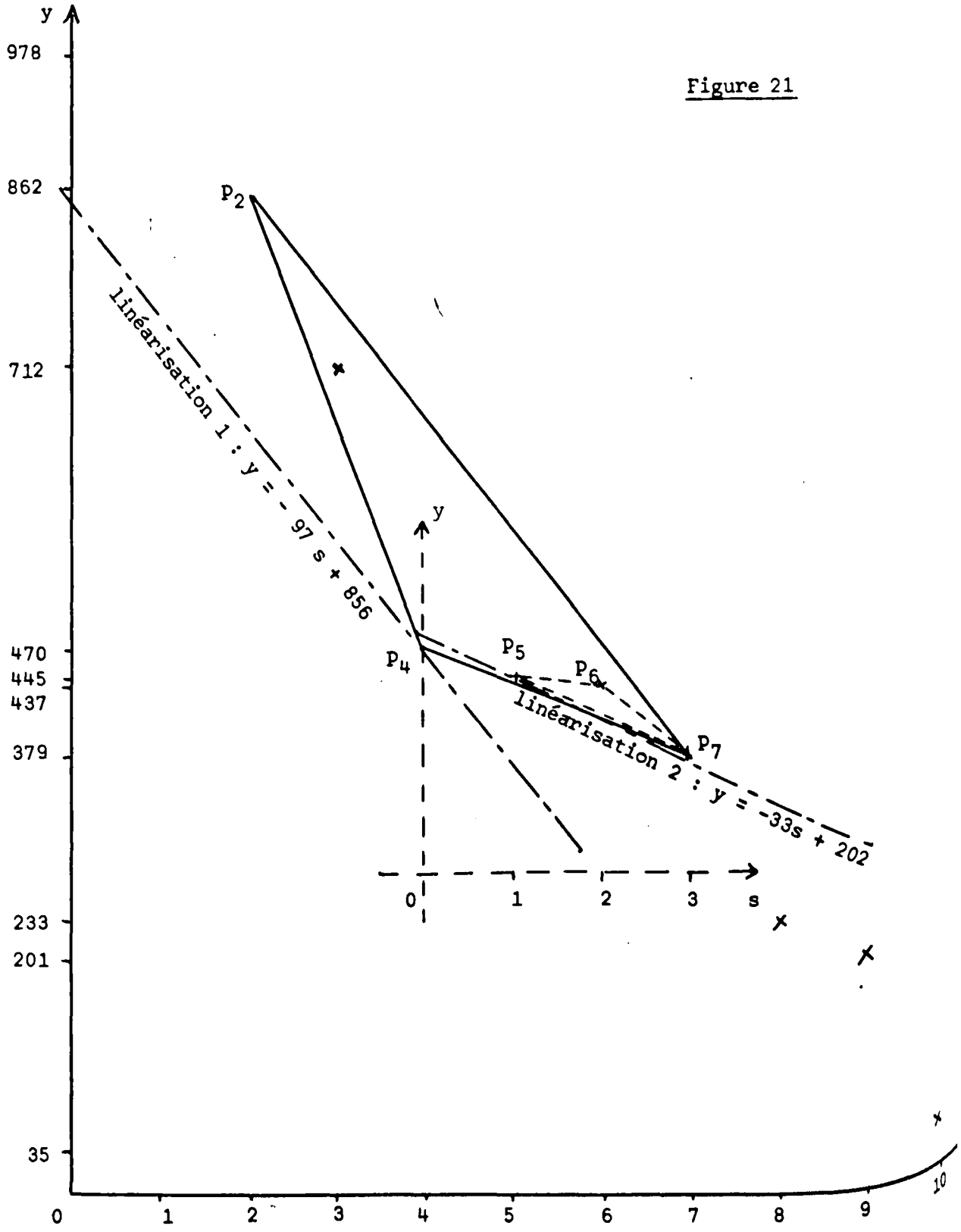
Problème 5 [Annexe] $1 \leq \Sigma x_j \leq 7$

Figure 20



Problème 5 [Annexe] $2 \leq \Sigma x_j \leq 7$

Figure 21



IV.5 - ALGORITHME D'ÉLIMINATION DÉFINITIVE DES VARIABLES

L'encadrement de $\sum_j x_j$, et la borne inférieure $\underline{v}(K)$ étant donnés par les étapes 1 à 6 de l'algorithme, la phase de réduction finale de notre méthode constitue l'étape 7 ; au déroulement de cette phase, explicité en § IV.1, il faut ajouter les deux précisions suivantes sur :

IV.5.1 - La linéarisation majorante

Sa détermination a été élaborée de manière tout à fait symétrique à celle de la linéarisation minorante détaillée en § IV.4.2.2.; il suffit, lorsque le point de linéarisation p_k de $\bar{H}_{\ell,u}$ a été sélectionné, de considérer la droite d'équation

$$y = \bar{\alpha}s + \bar{\beta} \text{ telle que } \bar{\alpha} = \lceil \theta_{\ell,u} \rceil \text{ et } \bar{\beta} = \lceil b(k) - k\,\theta_{\ell,u} \rceil$$

pour construire

$$(\emptyset LK) \qquad \left\lfloor \begin{array}{l} \max \sum_{j \in X_2} c_j x_j \\ \text{s.c.} \sum_{j \in X_2} (a_j - \bar{\alpha}) x_j \leq \bar{\beta} \\ x_j = 0 \text{ ou } 1 \ \forall j \in X_2 \end{array} \right.$$

IV.5.2 - La borne supérieure de $v(K \mid x_k = \varepsilon_k)$, $\varepsilon_k \in \{0,1\}$, $k \in X_2$

La résolution de $(\overline{\emptyset LK})$ par l'algorithme NKR fournit une partition de $X_2 = U \cup \{i\} \cup L$ rappelée au § IV.1 ; la borne supérieure utilisée dans l'algorithme mis au oeuvre pour la majorité des problèmes, est celle liée à la relaxation lagrangienne de $(\emptyset LK)$ associée à $\bar{\lambda} = c_i/a_i$, multiplicateur optimal de la contrainte $\sum_{j \in X_2} (a_j - \bar{\alpha}) x_j \leq \bar{\beta}$:

$$(RK(\bar{\lambda})) \max \sum_{j \in X_2} c_j x_j + \bar{\lambda}(\bar{\beta} - \sum_{j \in X_2} (a_j - \bar{\alpha}) x_j) \text{ s.c. } x_j = 0 \text{ ou } 1 \ \forall j \in X_2$$

Ainsi, étant donnés $k \in X_2$ et $\varepsilon_k = 1$ si $k \in U$, 0 si $k \in L$, l'évaluation par excès de l'étape 7 sera

$$\bar{v}(K \mid x_k = \varepsilon_k) = \lfloor v(RK(\bar{\lambda}) \mid x_k = \varepsilon_k) \rfloor .$$

Après avoir noté que

Propriété 11

$$|U| < \ell - 1 \Rightarrow v(\overline{\emptyset LK}) < \underline{v}(K)$$

Démonstration

En posant $\bar{C}_q \subset X_2$: $\begin{cases} \forall j \in \bar{C}_q \; c_j \geq c_h \quad \forall h \in X_2 \setminus \bar{C}_q \\ |\bar{C}_q| = q \leq |X_2| \end{cases}$

$$v(\overline{\emptyset LK}) = \sum_{j \in U} c_j + c_i \bar{x}_i \leq \sum_{j \in U} c_j + c_i \leq \sum_{j \in \bar{C}_q} c_j \text{ où } q = |U| + 1$$

$$\left.\begin{array}{l} \Rightarrow v(\overline{\emptyset LK}) < \sum_{j \in \bar{C}_{\ell-1}} c_j \\ \text{théorèmes 33 et 34} \end{array}\right\} \Rightarrow v(\overline{\emptyset LK}) < \underline{v}(K)$$

□

et qu'à l'évidence

Propriété 12

(*i*) *si* $|U| \in \{\ell-1, \ldots, u\}$
alors $v(RK(\bar{\lambda}) \mid \ell \leq \sum_{j \in X_2} x_j \leq u) = v(\overline{\emptyset LK})$

(*ii*) *si* $|U| \in \{\ell, \ldots, u-1\}$ *ou* $|U| = \ell-1$ *et* $k \in L$
alors $v(RK(\bar{\lambda}) \mid x_k = \varepsilon_k ; \ell \leq \sum_{j \in X_2} x_j \leq u) = v(RK(\bar{\lambda}) \mid x_k = \varepsilon_k)$

deux autres types de majorants de $v(K \mid x_k = \varepsilon_k)$, de valeurs plus faibles que $\lfloor v(RK(\bar{\lambda}) \mid x_k = \varepsilon_k) \rfloor$ peuvent être envisagés lorsque $|U| \geq u$:

a) *la valeur de la relaxation lagrangienne de* $(\emptyset LK)$ *dans laquelle est introduite la contrainte de borne (supérieure) sur la somme des variables*, c'est-à-dire

$$\lfloor v(RK(\bar\lambda) \mid x_k = \varepsilon_k \; ; \sum_{j\in X_2} x_j \le u)\rfloor = \lfloor v(RK(\bar\lambda) \mid x_k = \varepsilon_k) - Q\rfloor$$

avec $Q = \begin{cases} 0 & \text{si } k \in U \text{ et } |U| = u \text{ ou } u+1 \\ \sum_{j\in \underline{U}} (c_j - \bar\lambda a'_j) & \text{sinon} \end{cases}$

où $\begin{cases} \underline{U} \subset U \text{ est tel que} \begin{cases} \forall j \in \underline{U}\ c_j - \bar\lambda a'_j \le c_h - \bar\lambda a'_h \ \forall h \in L\setminus\underline{U} \\ |\underline{U}| = \begin{cases} |U| - u - 1 \text{ si } k \in U \text{ et } |U| \ge u+2 \\ |U| - u + 1 \text{ si } k \in L \end{cases} \end{cases} \\ a'_j = a_j - \bar\alpha \ \forall j \in X_2 \end{cases}$

Notes

(i) lorsque $|U| \ge \ell-1$,

$$v(RK(\bar\lambda) \mid x_k = \varepsilon_k \; ; \ell \le \sum_{j\in X_2} x_j \le u) = \bar\lambda\bar\beta + \sum_{j\in\bar U} (c_j - \bar\lambda a'_j)$$

avec $\bar U \subset U$:

$$\begin{cases} \forall j \in \bar U\ c_j - \bar\lambda a'_j \ge c_h - \bar\lambda a'_h \quad \forall h \in U\setminus\bar U \\ |\bar U| = \begin{cases} i \text{ si } |U| < u \\ u \text{ sinon} \end{cases} \end{cases}$$

(ii) lorsque $k \in U$ et $|U| = \ell - 1$,

$$v(RK(\bar\lambda) \mid x_k = \varepsilon_k \; ; \ell \le \sum_{j\in X_2} x_j) = v(RK(\bar\lambda) \mid x_k = \varepsilon_k) + \max_{j\in L} (c_j - \bar\lambda a'_j).$$

□

b) *la valeur de la relaxation lagrangienne de la relaxation du problème*

$$\begin{cases} \max \sum_{j \in X_2} c_j x_j \\ \text{s.c.} \sum_{j \in X_2} a'_j x_j \leq \bar{\beta} \qquad (81) \\ \sum_{j \in X_2} x_j \leq u \qquad (82) \\ x_k = \varepsilon_k \; ; \; 0 \leq x_j \leq 1 \; \forall j \in X_2 \backslash \{k\} \end{cases}$$

où $a'_j = a_j - \bar{\alpha} \quad \forall j \in X_2$

obtenue en remplaçant les contraintes (81) et (82) par une contrainte composite :

$$\begin{cases} \max \sum_{j \in X_2} c_j x_j \\ \text{s.c.} \sum_{j \in X_2} (a'_j + \gamma) x_j \leq \bar{\beta} + \gamma u \\ x_k = \varepsilon_k \; ; \; 0 \leq x_j \leq 1 \quad \forall j \in X_2 \backslash \{k\} \end{cases}$$

La détermination du multiplicateur γ a été réalisée en utilisant les résultats sur la contraction de contraintes qui permettent de remplacer le système

$$\begin{cases} a'x + y_1 = \bar{\beta} \\ e\,x + y_2 = u \\ e\,x \geq \ell \\ x \in V \; ; \; y_1, y_2 \in \mathbb{N} \end{cases}$$

par le système équivalent

$$\begin{cases} (a' + \gamma e)\, x + y_1 + \gamma y_2 = \bar{\beta} + \gamma u \\ e\,x \geq \ell \\ x \in V \; ; \; y_1, y_2 \in \mathbb{N} \end{cases}$$

Plus précisément, ce sont les résultats de [E2] et de [G13] qui ont été employés :

en posant

$$\underline{A}_q \subset X_2 : |\underline{A}_q| = q \leq |X_2| \text{ et } \forall j \in \underline{A}_q \; a'_j \leq a'_h \; \forall h \in X_2 \backslash \underline{A}_q$$

$$\bar{A}_q \subset X_2 : |\bar{A}_q| = q \leq |X_2| \text{ et } \forall j \in \bar{A}_q \; a'_j \geq a'_h \; \forall h \in X_2 \backslash \bar{A}_q$$

et en remarquant que

$$0 \leq y_1 \leq \bar{\beta} - \min \{a'x \mid ex \geq \ell \; ; \; x \in V\} = \bar{\beta} - \sum_{j \in \underline{A}_\ell} a'_j$$

$$0 \leq y_2 \leq u - \min \{ex \mid ex \geq \ell \; ; \; x \in V\} = u - \ell$$

les résolutions de

$$(P_1) \quad \left[\begin{array}{l} -\bar{\beta} + \max \; a'x + y_1 \\ \quad \text{s.c.} \quad ex \quad + y_2 \leq u - 1 \\ \quad\quad x \in V \; ; \; 0 \leq y_2 \leq u - \ell \end{array} \right.$$

et de

$$(P_2) \quad \left[\begin{array}{l} -\bar{\beta} + \min \; a'x + y_1 \\ \quad \text{s.c.} \quad ex \quad + y_2 \geq u + 1 \\ \quad\quad x \in V \; ; \; 0 \leq y_2 \leq u - \ell \end{array} \right.$$

aboutissent à

$$\gamma = \max \{v(P_1), -v(P_2)\} + 1$$

$$= \max \{ \sum_{j \in \bar{A}_{u-1}} a'_j - \sum_{j \in \underline{A}_\ell} a'_j, \; \bar{\beta} - \sum_{j \in \underline{A}_{\ell+1}} a'_j \} + 1$$

IV.6 - EXEMPLES NUMÉRIQUES

Deux exemples de déroulement de la méthode sont proposés : les problèmes 2 et 5 de l'annexe, pour chacun desquels sont envisagés les deux cas correspondant à l'utilisation, en étape 3, des algorithmes G_1 ou G_2 ; les temps sont fournis en millisecondes (mise en oeuvre sur UNIVAC 1110).

IV.6.1 - Problème 2 (Annexe)

(i) Utilisation de l'algorithme G_1 :

Temps

1.6 1 $\sum_{j=1}^{n} x_j \leq 11 = u$ (algorithme B_3)

1 $\ell = 4 \leq \sum_{j=1}^{n} x_j$ et $\underline{v}(K) = 274$ (algorithme B_1)

.4 2 aucune variable éliminée

3.8 3 l'algorithme G_1 fournit $\underline{v}(K) = 497$ et la solution $\underline{x}$ associé :
$\underline{x}_j = 1 \ \forall\, j \in U$, $0 \ \forall\, j \notin U$:
$U = \{7, 11, 12, 16, 1, 9, 8, 24, 20, 18\}$

avec
c_j : 64 56 32 45 75 86 35 8 58 38
a_j : 7 4 6 14 31 42 13 3 33 44 ; $a\underline{x} = 197$

1.2 4 $\ell = 7 \leq \sum_{j=1}^{n} x_j$ (algorithme B_2)

(Etant donné $U = \{1,7,9,21,23,26\}$ l'ensemble des indices des six plus grands éléments de la suite (c_j) :

$$\sum_{j\in U} c_j = 490 < \underline{v}(K) = 497 < \sum_{j\in U} c_j + \max_{j\notin U} c_j = 490 + c_{20} = 548)$$

.4 5 aucune variable éliminée

.6 6 l'algorithme B_4 fournit

$U = \{ 7,\ 11,\ 12,\ 16,\ 1,\ 9,\ 8,\ 24,\ 20,\ 21\}$

avec $\begin{cases} c_j : 64 & 56 & 32 & 45 & 75 & 86 & 35 & 8 & 58 & 93 \\ a_j : 7 & 4 & 6 & 14 & 31 & 42 & 13 & 3 & 33 & 60 \end{cases}$ et $c_{21}/a_{21} = \min_{j\in U} c_j/a_j$

tels que $\sum_{j\in U} a_j - a_{21} = 153 < b(10) = 207 < \sum_{j\in U} a_j = 213$

$\longrightarrow \sum_{j\in U\setminus\{21\}} c_j + c_{21}(207 - 153)/a_{21} = 459 + 93 \times 54/60 = 542.7$

$\longrightarrow \bar{Z} = \lfloor 542.7 \rfloor = 542 > \underline{v}(K) \longrightarrow$ bien que $|U| = 10$, la valeur de u ne peut être diminuée.

7 phase de réduction

7.1 $\bar{H}_{7,11} = \bar{M}_{7,11} = \{p_7, p_8, p_9, p_{10}, p_{11}\}$

linéarisation majorante : $y = -11s + 315$

(p_9 est le point le plus éloigné de $(p_7 p_{11})$)

7.2 - 7.8

7.2 la résolution de la relaxation de

(ØLK) $\max \sum_{j=1}^{n} c_j x_j$ s.c. $\sum_{j=1}^{n} (a_j+11)x_j \leq 315$; $x_j = 0$ ou 1 $\forall$ j

fournit :

$U = \{ 7,\ 11,\ 12,\ 16,\ 1,\ 9,\ 8,\ 20,\ 21\}$

avec $\begin{cases} e_j : 64 & 56 & 32 & 45 & 75 & 86 & 35 & 58 & 93 \\ a_j : 7 & 4 & 6 & 14 & 31 & 42 & 13 & 33 & 60 \end{cases}$

et $i = 26$ ($c_{26} = 77$; $a_{26} = 65$) : $\sum_{j\in U} (a_j+11) = 309 < 315 < 309+65+11$

$\longrightarrow v(\overline{\text{ØLK}}) = \sum_{j\in U} c_j + 77 \times (315-309)/76 = 550.08$

7.3 $\lfloor v(\overline{\text{ØLK}}) \rfloor > \underline{v}(K) \longrightarrow$ pas de conclusion sur l'optimalité de $\underline{v}(K)$

7.4 $\underline{v}(\text{ØLK}) = \sum_{j\in U} c_j = 544 > \underline{v}(K) \longrightarrow \underline{v}(K) = 544$

($\sum_{j\in U} a_j = 210 < b(9) = 221$)

7.5 élimination définitive et fixation à 1 des 9 variables d'indices $j \in X_1 = \{12,11,7,16,8,21,1,9,20\}$ (les coûts réduits à l'optimum de $(\overline{\emptyset LK})$, associés à ces variables sont tous supérieurs à $10.66 > v(\overline{\emptyset LK}) - \underline{v}(K) = 6.08$).

$\longrightarrow \ell = 9+1 = 10$; la nouvelle valeur de $b(\ell)$ est alors

$b(10) - \sum_{j \in X_1} a_j = 207 - 210 = -3 \longrightarrow v(K) = 544$;

la solution associée à $\underline{v}(K)$ est optimale (toutes les variables d'indices $j \notin X_1$ sont nulles). □

(ii) Utilisation de l'algorithme G_2 :

Les seules modifications par rapport au (i), interviennent aux étapes 3, 6, 7-1, 7-2 et 7-5 :

3 l'algorithme G_2 fournit dorénavant - en un temps identique à G_1 - la valeur optimale de (K) : $\underline{v}(K) = 544$ et la solution $\underline{x}$ associée : $\underline{x}_j = 1 \ \forall j \in U, 0 \ \forall j \notin U$:

$U = \{ 12, 11, 7, 16, 1, 9, 20, 8, 21\}$

avec $\begin{cases} c_j : 32 & 56 & 64 & 45 & 75 & 86 & 58 & 35 & 93 \\ a_j : 6 & 4 & 7 & 14 & 31 & 42 & 33 & 13 & 60 \end{cases}$; $\underline{ax} = 210$

6 la valeur $\bar{z} = 542$ obtenue par l'algorithme B_4 est alors inférieure à $\underline{v}(K) = 544 \longrightarrow u = |U| - 1 = 9$

7.1 $\bar{H}_{7,9} = \bar{M}_{7,9} = \{p_7, p_8, p_9\}$ (temps : .2 ms)

linéarisation majorante en p_8 : $y = -5s + 265$ (temps : .2 ms)

7.2 la résolution de la relaxation de

$$(\emptyset LK) \quad \max \sum_{j=1}^{n} c_j x_j \ \text{s.c.} \sum_{j=1}^{n} (a_j+5) \leq 265 \ ; \ x_j = 0 \text{ ou } 1 \ \forall j$$

fournit la même partition de $\{1,\ldots,n\}$ que dans le (i) avec cependant une valeur différente de $\lfloor v(\overline{\emptyset LK}) \rfloor = \lfloor 544+77\times(265-255)/65 \rfloor = 555$

IV.6.2 - Problème 5 (Annexe)

Bien que l'algorithme G_1 aboutisse à une valeur de $\underline{v}(K)$ inférieure à celle donnée par l'algorithme G_2, l'algorithme B_4 permettra, en étape 6, de "retrouver" la valeur obtenue par G_2 ; ce qui impliquera un déroulement identique de toute l'étape 7 pour les deux variantes de notre méthode dont les temps de calcul globaux seront sensiblement différents : temps avec G_2 = 67% temps avec G_1 :

(i) Utilisation de l'algorithme G_1 :

Temps

2.4 1 $\quad \sum_{j=1}^{n} x_j \leq 7 = u$ (algorithme B_3)

1.8 $\quad \ell = 2 \leq \sum_{j=1}^{n} x_j$ et $\underline{v}(K) = 300$ (algorithme B_1)

1 2 les 10 variables d'indices $j \in E = \{12,25,32,36,40,53,63,74,79,96\}$ sont éliminées et fixées à 0 (les valeurs des a_j associés sont supérieures à $b(2) = 862$) ; $X_2 \equiv \{1,\ldots,n\} \setminus E$ ($\Longrightarrow |X_2| = 90$)

13.6 3 l'algorithme G_1 fournit $\underline{v}(K) = 1094$ et la solution $\underline{x}$ associée : $x_j = 1 \ \forall j \in U$, $0 \ \forall j \notin U$:

$U = \{ 42, 28, 65, 59, 51, 70\}$

avec
c_j :	221	245	246	256	115	11
a_j :	22	50	120	150	34	61 ; $a\underline{x} = 437$

2 4 $\quad \ell = 4 \leq \sum_{j \in X_2} x_j$ (algorithme B_2)

(Etant donné $U = \{2,52,58\}$ l'ensemble des indices des 3 plus grands éléments de la suite (c_j) :

$$\sum_{j \in U} c_j = 892 < \underline{v}(K) = 1094 < \sum_{j \in U} c_j + \max_{j \notin U} c_j = 892 + c_{82} = 1183)$$

1.2 5 48 nouvelles variables sont éliminées et fixées à 0 (les valeurs des a_j correspondants appartiennent à $]b(4),b(2)]$)
$\Longrightarrow X_2 = \{42,28,65,59,51,70,8,31,3,6,86,2,88,10,83,34,17,98,68,16,$
$21,39,23,33,85,26,67,73,15,30,71,66,64,84,11,57,94,55,95,9,50,46\}$
$(|X_2| = 42)$.

1.8 6 l'algorithme B_4 fournit

$U = \{ 42,\ 28,\ 65,\ 3,\ 51,\ 31,\ 86\}$

avec $\begin{cases} c_j : 221 & 245 & 246 & 151 & 115 & 211 & 226 \\ a_j : 22 & 50 & 120 & 66 & 34 & 34 & 125 \end{cases}$ et $c_{86}/a_{86} = \min\limits_{j\in U} c_j/a_j$

tel que $\sum\limits_{j\in U} a_j - a_{86} = 376 < b(7) = 379 < \sum\limits_{j\in U} a_j = 501$

$\longrightarrow \sum\limits_{j\in U\setminus\{86\}} c_j + c_{86}(379-376)/a_{86} = 1189+226\times3/125 = 1194.4$

avec
(i) $\bar{z} = 1194 > \underline{v}(K) = 1094 \Longrightarrow$ la valeur de u n'est pas diminuée
(ii) $\underline{z} = \sum\limits_{j\in U\setminus\{86\}} c_j = 1189$ car $|U\setminus\{86\}| = 6$ et $\sum\limits_{j\in U\setminus\{86\}} a_j = 376 < b(6) = 437$

$\longrightarrow \underline{v}(K) = 1189$

2.2 répétition des étapes 4 et 5 :

* $\ell = 5 \leq \sum\limits_{j\in X_2} x_j$ (algorithme B_2)

(étant donné $U = \{2,10,17,88\}$ l'ensemble des indices des quatre plus grands éléments de la suite $(c_j)_{j\in X_2}$:

$\sum\limits_{j\in U} c_j = 1156 < \underline{v}(K) = 1189 < \sum\limits_{j\in U} c_j + \max\limits_{j\notin U} c_j = 1156 + c_{59} = 1412$)

* 3 nouvelles variables d'indices $j \in E = \{55,57,95\}$ sont éliminées et fixées à 0 ; $X_2 \equiv X_2 \setminus E$ $(\Longrightarrow |X_2| = 39)$.

7 phase de réduction

.2 **7.1** $\bar{H}_{5,7} = \bar{M}_{5,7} = \{p_5, p_6, p_7\}$

.2 linéarisation majorante en p_6 : $y = -33s + 635$

3 **7.2 à 7.8**

7.2 la résolution de la relaxation de

$$(\emptyset LK) \max \sum_{j\in X_2} c_j x_j \blacktriangle c. \sum_{j\in X_2} (a_j+33)x_j \leq 635 \ ; \ x_j = 0 \text{ ou } 1 \quad \forall\, j \in X_2$$

fournit

$$U = \{\ 42,\ \ 28,\ \ 31,\ \ 65,\ \ 51,\ \ 3\}$$

$$\text{avec} \begin{cases} c_j : & 221 \quad 245 \quad 211 \quad 246 \quad 115 \quad 151 \\ a_j : & 22 \quad 50 \quad 84 \quad 120 \quad 34 \quad 66 \end{cases}$$

et $i = 76$ ($c_{76}=226$; $a_{76}=125$) : $\sum_{j\in U} (a_j+33) = 574<635<574+125+33$

$$\Longrightarrow v(\overline{\emptyset LK}) = \sum_{j\in U} c_j + 226\times(635-574)/158 = 1189+226\times 61/158 = 1276.25$$

7.3 et 7.4 $\lfloor v(\overline{\emptyset LK})\rfloor > \underline{v}(K) = \underline{v}(\emptyset LK) \Longrightarrow$ pas de test d'arrêt ni de modification de $\underline{v}(K)$.

7.5 élimination définitive et fixation à 1 des 2 variables d'indices $j \in X_1 = \{28,42\} \Longrightarrow \sum_{j\in X_1} c_j = 466$; $\sum_{j\in X_1} a_j = 72$; $X_2 \equiv X_2 \setminus X_1$;

$s = s - |X_1| = s-2 \quad s=5,6,7 \Longrightarrow \ell = 3$ et $u = 5$;

$b(3) = 445-72 = 373$; $b(4) = 437-72 = 365$; $b(5) = 379-72 = 307$.

7.6 élimination définitive et fixation à 0 de 30 variables d'indices $j \in L = X_2 \setminus (U \cup \{i\})$:

* 6 d'entre elles, d'indices $j \in \{15,17,33,64,73,84\}$, sont telles que $a_j > b(3)$;

* les 24 autres sont telles que $\bar{v}(\emptyset LK \mid x_j = 1) < \underline{v}(K)$ (utilisation des coûts réduits à l'optimum de $(\overline{\emptyset LK})$).

Le problème initial est donc réduit à 7 variables :

$$466 + \max\ 151x_3 + 246x_{65} + 115x_{51} + 211x_{31} + 226x_{76} + 298x_2 + 256x_{59}$$
$$\text{s.c.}\quad 66x_3 + 120x_{65} + 34x_{51} + 84x_{31} + 125x_{76} + 231x_2 + 150x_{59} \le b\Big(\sum_{j\in X_2} x_j\Big)$$
$$3 \le x_3 + x_{65} + x_{51} + x_{31} + x_{76} + x_2 + x_{59} \le 5$$
$$x_j = 0 \text{ ou } 1 \qquad j = 3, 65, 51, 31, 76, 2, 59$$

avec b(3) = 373, b(4) = 365 et b(5) = 307

(ii) Utilisation de l'algorithme G_2 :

Les modifications par rapport au (i) interviennent aux étapes 3, 4 et 5 :

3 l'algorithme G_2 fournit - en un temps moitié moindre à celui de G_1 (7 ms) - la valeur $\underline{v}(K) = 1189$ et la solution $\underline{x}$ associée : $\underline{x}_j = 1\ \forall\ j \in U$, $0\ \forall\ j \notin U$:

$U = \{42, 28, 31, 65, 51, 3\}$

avec $\begin{cases} c_j : 221 & 245 & 211 & 246 & 115 & 151 \\ a_j : 22 & 50 & 84 & 120 & 34 & 66 \end{cases}$; $\underline{a}\underline{x} = 376$

4 l'algorithme B_2 aboutit directement (sans utilisation préalable de l'algorithme B_4) à $\ell = 5 \le \sum_{j\in X_2} x_j$

(étant donné $U = \{2,52,58,82\}$ l'ensemble des indices des quatre plus grands éléments de la suite $(c_j)_{j\in X_2}$:

$$\sum_{j\in U} c_j = 1183 < \underline{v}(K) = 1189 < \sum_{j\in U} c_j + \max_{j\notin U} c_j = 1189 + c_{10} = 1473)$$

5 51 variables sont éliminées et fixées à 0 (elles correspondent à celles éliminées en deux fois par la "variante G_1" (répétition des étapes 4 et 5 après l'étape 6)).

Note : les tests de fin d'étape 6 étant négatifs, il n'y a plus répétition des étapes 4 et 5 avec la "variante G_2".

V - EXPERIENCES NUMERIQUES

La méthode de réduction de (K) a été mise en oeuvre sur UNIVAC 1110 en considérant

- d'une part - § V.1 - les cinq problèmes tests de 30, 50 et 100 variables utilisés dans [EK9] (voir les caractéristiques en annexe)

- d'autre part - § V.2 - des problèmes de 30, 50, 100 et 1000 variables tirés au hasard suivant la loi uniforme.

Pour chaque série de problèmes, n désignera leur taille et $b = \max \{j \in \{1,\ldots,n\} \mid b(j) > 0\}$; tous les temps seront donnés en millisecondes.

V.1 - PROBLEMES TIRES DE [EK9]

Le tableau 46 indique pour chacun des cinq problèmes les résultats significatifs obtenus en :

- étapes 1 et 2 : encadrement de $\sum_j x_j$; borne inférieure de v(K) atteinte par l'algorithme B_1 ; nombre de variables après élimination ($|X_2|$) ; temps d'exécution.

- étape 3 : bornes inférieures de v(K) atteintes par les algorithmes G_1 et G_2 ; temps d'exécution respectifs (avec pour G_2 l'indication supplémentaire entre parenthèses - du temps partiel nécessaire aux constructions d'enveloppes convexes).

- étapes 4, 5 et 6 : encadrement final de $\sum_j x_j$; nombre de variables après élimination ; temps d'exécution.

* Note : (problème 5) : dans la variante de la méthode utilisant l'algorithme G_1, l'algorithme B_4 permet d'aboutir à $\underline{v}(K) = 1189$, valeur obtenue directement par l'algorithme G_2 dans l'autre variante.

□

- étape 7-1 : temps nécessaire à la détermination de l'enveloppe convexe de $\bar{M}_{\ell,u}$ puis de la linéarisation.

- étapes 7-2 à 7-8 : borne supérieure de v(K) donnée par linéarisation majorante ; nombres de variables éliminées à 1 ($|X_1|$) et à 0($|X_0|$) ; nombre de variables restant après élimination ; temps d'exécution.

* Note : (problème 2) : dans la variante de la méthode utilisant l'algorithme G_1, la résolution de la relaxation de la linéarisation majorante permet l'amélioration de la borne inférieure de v(K) qui atteint ainsi 544, valeur obtenue directement par l'algorithme G_2 dans l'autre variante.

□

La dernière colonne de ce tableau 46 fournit enfin les temps globaux de réduction pour chacune des deux variantes de la méthode (utilisation de G_1 ou G_2).

Le tableau 47 résume les résultats de la "variante G_2" détaillés dans le tableau 46 ; il fournit successivement pour chaque problème :

- la borne inférieure de v(K)
  * note : seule celle du problème 5 n'atteint pas v(K) dont la valeur est 1264

□

- l'encadrement de $\sum_j x_j$ avec l'indication supplémentaire du nombre de variables mises à 1 à l'optimum ($\sum_j x_j^*$)

- le nombre de variables éliminées à 1 ($|X_1|$) et à 0 ($|X_0|$)

- le nombre de variables non éliminées par la méthode de réduction

- le temps global de réduction par notre méthode

- les temps globaux de résolution de (K) par la méthode de [EK9] :
  . d'une part, précédée de notre phase de réduction (temps global)
  . d'autre part, sans phase initiale de réduction (temps de [EK9]).

n	m	Etapes 1 et 2					Etape 3		Etapes 4, 5, 6				Etape 7-1	Etapes 7-2 à 7-8					Temps
		ℓ	u	$\underline{v}(K)$	$\lvert X_2\rvert$	Temps	$\underline{v}(K)$	Temps	ℓ	u	$\lvert X_2\rvert$	Temps	Temps	$\tilde{v}(K)$	$\lvert X_1\rvert$	$\lvert X_0\rvert$	$\lvert X_2\rvert$	Temps	Total
30	10	4	6	274	30	2.4	G_1 358	2.2	5	6	30	2	.4	362	5	24	1	1.6	8.6
							G_2 358	3.6 (.4)											10
30	30	4	11	274	30	3	G_1 497	3.8	7	11	30	2.2	.6	550*	9	21	0	1.6	11.2
							G_2 544	3.8 (.8)	7	9		2.6	.4	555				1.8	11.6
30	30	6	14	426	30	2.6	G_1 647	4.8	9	14	30	2	.4	653	10	18	2	2.2	12
							G_2 647	2.8 (.4)											10
50	10	3	6	575	39	3	G_1 1256	3	5	6	25	3	.4	1330	5	44	1	2.6	12
							G_2 1256	3.8 (.4)				2.6							12.4
200	10	2	7	300	90	5.2	G_1 1094	13.6	5	7*	39	7.2	4	1278	2	92	6	25.8	52.5
							G_2 1189	7 (.6)	5	7		4							42.4

TABLEAU 46

n	m	$\underline{v}(K)$	ℓ	Σx_j^*	u	$\lvert X_1\rvert$	$\lvert X_0\rvert$	$\lvert X_2\rvert$	Temps	Temps global	Temps de [EK9]
30	10	358	5	6	6	5	24	1	10	10	331
30	30	544	7	9	9	9	21	0	11.6	11.6	1283.2
30	30	647	9	13	14	10	18	2	10	10	1501.2
50	10	1256	5	6	6	5	44	1	12.4	12.4	318.8
200	10	1189*	5	6	7	2	91	6	42.4	48.8	2324.4

TABLEAU 47

V.2 - PROBLÈMES TIRÉS AU HASARD

La méthode a également été testée sur 415 problèmes de n = 30, 50, 100 et 1000 variables, dont les données - entières positives - ont été tirées au hasard suivant la loi uniforme dans des intervalles dont les bornes supérieures sont répertoriées dans le tableau suivant (colonnes c, a, b) :

Tableau	Série	n	NP	c	a	b	m
49	1 2	30	10	100	100	300 600	10, 20, 30
50	3 4	30	10	200	200	300 600	10, 20, 30
51	5 6	50	10	300	1000	1000 2000	10, 20, 30 40, 50
52	7 8 9	100	10	300	1000	1000 5000 10000	10, 30, 50 70, 100
54	10 11 12	1000	5	300	1000	1000 10000 50000	10,100,500

Les 12 séries de problèmes ont été construites comme suit : pour chaque taille n donnée

- d'une part, NP couples $(c,a) \in \mathbb{N}^n \times \mathbb{N}^n$ ont été tirés au hasard suivant les caractéristiques décrites ci-dessus

- d'autre part, pour chaque série associée à n, le tirage au hasard de n nombres $b_1, b_2, \ldots, b_n$, bornés supérieurement par la valeur correspondante de la colonne b, a permis la construction de différentes fonctions b : à

chaque valeur de m (indiquée en colonne m du tableau précédent) a été associée une fonction b définie sur $\{1,2,\ldots,m\}$ et à valeurs $b_{i_1}, b_{i_2}, \ldots, b_{i_m}$, correspondant aux m premiers nombres $b_1, b_2, \ldots, b_m$ rangés par ordre décroissant.

Ainsi, chaque série $i \in \{1,\ldots,12\}$ correspond à NP × m_i problèmes tests, où m_i désigne le nombre d'éléments de la colonne m associée :

Exemple : la série 1 est un ensemble de 30 problèmes ; à chacune des $m_1=3$ valeurs de m (10,20 et 30) correspond une fonction b unique - à valeur dans $]0,300[$ - qui est associée à NP = 10 couples (c,a) distincts dont les éléments sont à valeurs dans $]0,100[\times]0,100[$. □

Pour chaque série de problèmes caractérisée par la borne supérieure des valeurs de la fonction b associée, et pour chaque valeur de m considérée, les tableaux 49, 50, 51, 52 et 54 rassemblent les différentes valeurs moyennes suivantes - NP problèmes sont testées - en colonnes :

Alg. G_1 (resp. Alg.G_2) : valeur et temps fournis par l'algorithme glouton G_1 (resp. G_2, avec l'indication supplémentaire du temps partiel pour les constructions d'enveloppes convexes)

$\underline{v}(K)$ (resp. $\bar{v}(K)$) : borne inférieure (resp. supérieure) obtenue en fin d'algorithme

$(\bar{v}(K) - \underline{v}(K)) / \underline{v}(K)$: pourcentage de cet écart relatif, borne supérieure sur la précision de notre méthode

ℓ (resp. u) : borne inférieure (resp. supérieure) finale de $\sum_{j=1}^{n} x_j$

$|X_2|$: nombre de variables non éliminées avant le déroulement de l'étape 7

Temps étapes 1 à 7-1 : temps nécessaire au déroulement des étapes 1 à 7-1 inclus, pour la variante G_2 de notre méthode

$|X_1|$, $|X_0|$, $|X_2|$: nombres respectifs de variables fixées à 1, à 0 et non éliminées en fin d'étape 7.

Temps réduction : temps nécessaire au déroulement des étapes 7-2 à 7-8

Temps total : somme du temps des étapes 1 à 7-1 (avec la variante G_2) et de celui des étapes 7-2 à 7-8.

Les tableaux 48 et 53 permettent une comparaison des temps de calcul. Pour chacune des séries concernées on a en colonnes :

Temps global (resp. Temps de [EK9]) : le temps global de résolution de (K) par la méthode de [EK9] avec (resp. sans) phase initiale de réduction

Note

Pour certains problèmes, les temps d'exécution de la méthode [EK9] sans phase de réduction, étant prohibitifs, la résolution de (K) a dû être interrompue dans les limites indiquées ; les moyennes fournies sont alors dissociées : par exemple, pour la série 4 (n=30, m=10) : l'exécution de 2 problèmes a été interrompue après 10 secondes, alors que leur résolution avec la phase initiale de réduction a nécessité une valeur moyenne de 49.3 ms ; les deux autres moyennes indiquées concernent les 8 autres problèmes.

□

		SERIES 1 et 2		SERIES 3 et 4	
b	m	Temps global	Temps de [EK9]	Temps global	Temps de [EK9]
300	10	1 514.26	4 535.36	45.27	1 064.02
	20	20.08 (5 - 36.72)	2 779.52 (>10000)	31.48	3 144.84
	30	28.25 (6-181.8)	2 763.75 (>10000)	43.36	2 476.4
600	10	858.42	3 718.02	383.51 (2 - 49.3)	1 138.71 (>10000)
	20	25.40 (5-769.45)	4 813.48 (>10000)	505.95 (6-2329)	2 716.65 (>10000)
	30	38.44 (5-142.8)	2 194.4 (>10000)	27.87 (7-106)	932.97 (>10000)

Tableau 48

Alg. G_1		Alg. G_2		$\underline{v}(K)$	$\bar{v}(K)$	$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	ℓ	u	$\lvert X_2\rvert$	Temps étapes 1 à 7-1	$\lvert X_1\rvert$	$\lvert X_0\rvert$	$\lvert X_2\rvert$	Temps réduction	Temps Total
$\underline{v}(K)$	Temps	$\underline{v}(K)$	Temps												
493.9	3.2	503.7	3.38 (.44)	507.2	548.2	8.08	6.2	7.7	30	9.28	1.2	12.5	16.3	2.80	12.08
643.5	3.65	641.2	3.18 (.40)	650	673.6	3.63	7.9	9.9	30	9.62	4.6	15.9	10.3	2.82	12.44
683.7	3.98	682.9	3.22 (.50)	689.7	709.5	2.90	8.6	10.5	30	9.56	5.6	15.2	9.2	2.74	12.30
813.1	3.32	635	2.78 (.20)	646.4	677.3	4.78	8.3	8.9	30	8.72	3.5	14	12.5	3.06	11.78
786.2	4.26	817.9	3.58 (.40)	832.9	890.3	6.89	10.8	11.6	30	9.40	3.5	7.7	18.8	2.44	11.64
947.8	4.76	961.6	3.70 (.62)	965.7	983	1.79	12.8	14.6	30	10.36	8.9	10.7	10.4	2.78	13.14

TABLEAU 49 (30 VARIABLES)

LILLE

Alg. G_1		Alg. G_2		$\underline{v}(K)$	$\bar{v}(K)$	$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	ℓ	u	$\lvert X_2\rvert$	Temps étapes 1 à 7-1	$\lvert X_1\rvert$	$\lvert X_0\rvert$	$\lvert X_2\rvert$	Temps réduction	Temps Total
$\underline{v}(K)$	Temps	$\underline{v}(K)$	Temps												
839.2	2.56	846.1	2.60 (.34)	847.7	920.9	8.64	5.2	6.4	29.7	8.46	2.5	17.6	9.9	3.10	11.56
995.4	3.46	994.7	3.02 (.46)	1001	1043.1	4.20	6.2	7.8	30	8.50	4.6	18.7	6.7	2.40	10.90
1002	3.40	1000.6	2.96 (.52)	1016.6	1066.6	4.92	6.4	8.1	30	8.50	4.1	18.3	7.6	2.52	11.02
1089	3.08	1103.1	2.76 (.32)	1111	1203.2	8.30	6.8	7.8	30	8.96	1.6	12.1	16.3	3.06	12.02
1334.5	3.94	1342.2	2.96 (.48)	1358.9	1443.8	6.25	8.4	10.5	30	9.20	3.2	11.5	15.3	2.96	12.16
1447.3	4.26	1453.7	3.16 (.62)	1458.4	1508.8	3.46	9.3	11.5	30	8.76	5.8	14.6	9.6	2.66	11.42

TABLEAU 50 (30 VARIABLES)

Alg. G_1		Alg. G_2		$\underline{v}(K)$	$\bar{v}(K)$	$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	ℓ	u	$\lvert X_2\rvert$	Temps étapes 1 à 7-1	$\lvert X_1\rvert$	$\lvert X_0\rvert$	$\lvert X_2\rvert$	Temps réduction	Temps Total
$\underline{v}(K)$	Temps	$\underline{v}(K)$	Temps												
1108.5	3.04	1129.5	3.06 (.46)	1132.6	1218.8	7.61	4.6	6.1	28.4	10.30	3.2	40.7	6.1	3.46	13.76
1299.2	3.88	1342.2	3.80 (.52)	1346.7	1419.8	5.43	5.6	7.4	34.5	12.10	4	38.9	7.1	4.64	16.74
1389.3	5.14	1389.3	4.24 (.64)	1408.8	1480.9	5.12	5.5	7.9	38.8	12.40	4.4	39	6.6	5.08	17.48
1548.5	5.06	1499.5	4.16 (.58)	1504.5	1549.2	2.97	6	8.3	43.7	13.06	6.1	39.3	4.6	5.84	18.90
1527.7	5.16	1525.7	4.58 (.66)	1529	1598.5	4.54	6.2	8.7	45.3	14.26	5.7	38.5	5.8	6.16	20.40
1424.9	4.90	1433.6	4.32 (.52)	1442.1	1543.8	7.05	5.7	7.2	47.9	12.68	3.3	34.4	12.3	4.18	16.88
1810	5.90	1810	4.54 (.72)	1822.3	1884.1	3.39	7.3	9.9	50	13.04	5.8	34.2	10	4.74	17.78
1914.1	6.52	1924.7	5.24 (.98)	1936.7	2013.9	3.97	7.7	10.4	50	12.62	5.9	33.9	10.2	3.28	15.90
2017.4	6.12	2022.6	4.84 (.94)	2029.4	2083.8	2.68	8.1	11.1	50	13.36	6.8	35.8	7.4	4.76	18.12
2109.8	7.02	2109.8	4.68 (1.06)	2119	2174.3	2.61	8.5	11.8	50	12.46	7.1	34.2	8.7	3.94	16.40

TABLEAU 51 (50 VARIABLES)

b	m	Alg. G_1 $\underline{v}(K)$	Alg. G_1 Temps	Alg. G_2 $\underline{v}(K)$	Alg. G_2 Temps	$\underline{v}(K)$	$\bar{v}(K)$	$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	ℓ	u	$\lvert X_2 \rvert$	Temps étapes 1 à 7-1	$\lvert X_1 \rvert$	$\lvert X_0 \rvert$	$\lvert X_2 \rvert$	Temps réduction
	10	1248.2	10.30	1261.8	7.06 (.48)	1285.5	1430.8	11.30	4.8	6.5	49.5	19.28	1	78.2	20.8	5.64
	30	2065	13.88	2099.6	8.56 (.80)	2099.6	2166.8	3.20	7.9	11	83.3	23.70	6.8	82.6	10.6	13.54
1000	50	2178.2	13.16	2178.2	8.52 (1)	2200	2253.8	2.45	8.3	12.5	86	24.38	8.6	83.9	7.5	11.70
	70	2253.7	15.78	2272.4	8.20 (.92)	2294	2330.7	1.60	8.6	12.7	92.8	26.18	8.7	84.7	6.6	15.90
	100	2300.5	14.66	2293.1	8.40 (.92)	2293.4	2341.9	2.11	8.6	13.2	93.5	25.82	9	84.7	6.3	15.74
	10	1922.8	9.95	1927.4	8.12 (.34)	1942.3	2103.5	8.30	7.3	8.6	97.4	22.66	1.8	70.9	27.3	9.66
	30	3502.3	18.78	3521.9	9.68 (.80)	3543.9	3656.4	3.17	13.2	16.8	100	22.64	3.2	61.1	35.7	5.90
3000	50	4017.8	22.20	4027.8	8.24 (.94)	4031.1	4189.4	3.93	15.2	20.6	100	22.86	3.2	48	48.8	9.72
	70	4544.6	24.22	4601	9.32 (1.3)	4617.6	4700.4	1.79	17.5	23.5	100	23.70	10.4	61.2	28.4	6.54
	100	4753.3	28.72	4770.2	9 (1.36)	4822.8	4877.6	1.13	18.2	26.6	100	25.14	15.7	65.6	18.7	8.02
	10	2152.7	10.54	2184.4	8.24 (.38)	2192.3	2395.8	9.28	8.2	9.6	100	24.78	.1	61.3	38.6	11.1
	30	4112.9	19.42	4102.7	9.48 (.70)	4120.6	4334.5	5.39	15.4	20.1	100	24.54	2.5	23.2	74.3	6.14
10000	50	4991.3	24.74	5075.5	10.66 (.92)	5085.7	5166.8	1.59	19.6	24.3	100	25.28	8.4	56.4	35.2	7.62
	70	5734.2	29.34	5879.4	9.60 (1.14)	5884.1	5997.9	1.93	22.9	29.6	100	25.46	8.7	38.9	52.4	8.20
	100	6294	31.42	6318.1	9.96 (1.44)	6323.1	6455.8	2.10	24.9	33.2	100	25.72	10.8	39.4	49.8	7.54

TABLEAU 52 (100 VARIABLES)

SERIES 5 et 6				SERIES 7 et 8			
b	m	Temps global	Temps de [EK9]	b	m	Temps global	Temps de [EK9]
	10	20.2	306.48		10	1 319.68	13 484.26
	20	21.96	1 081.71		20	281.34	>15 000
1000	30	41.50	2 206.86	1000	50	84.16	>15 000
	40	20.16 (1 - 18.14)	2 344.69 (>12000)		70	626.96	>15 000
	50	28.67 (1 - 19.4)	3 208.42 (>12000)		100	65.90	>15 000
	10	337.46 (3-331)	4 372.57 (>12000)		10	326.40	>15 000
	20	22.07 (7-412)	8 171.93 (>12000)		30	45.69	>15 000
2000	30	38.0 (9-304)	4 640.6 (>12000)	5000	50	50.60	>15 000
	40	17.4 (9-178)	5 915 (>12000)		70	1 181.56	>15 000
	50	153	>12000		100	207.89	>15 000

Tableau 53.

Alg. G_1		Alg. G_2		$\underline{v}(K)$	$\bar{v}(K)$	$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	ℓ	u	$\lvert X_2 \rvert$	Temps étapes 1 à 7-1	$\lvert X_1 \rvert$	$\lvert X_0 \rvert$	$\lvert X_2 \rvert$	Temps réduction	Temps Total
$\underline{v}(K)$	Temps	$\underline{v}(K)$	Temps												
7008.2	78.21	2174.2	48.16 (.56)	2185	3317	51.81	8	8.4	344	170.12	0	783.6	216.4	36.64	206.76
7388.2	553.68	7384.4	86.44 (2.92)	7384.4	7936.4	-	25.4	40	961	247.72	34.2	964.6	1.2	191.48	439.12
7388.2	554	7384.4	87.60 (2.92)	7384.4	7336.4	-	25.4	40	961	248.92	34.2	964.6	1.2	191.40	440.32
2734.8	91.44	2700	57.28 (.28)	2725.2	4823.2	76.99	9.8	10	493.4	169.20	0	648.8	351.2	82.72	251.92
20436.6	969.60	20620.6	71.12 (5.52)	20680.8	21665.4	4.76	72	99	1000	207.92	50	530	420	102	309.92
21633.8	1184.68	21651.2	98.36 (6.36)	21655.6	21695.8	.19	75.6	110	1000	226.80	74.8	864.6	60.6	53.56	280.36
2765.8	104.68	2707.4	62.88 (.64)	2791.6	11836.6	324	10	10	1000	254.32	0	0	1000	331.88	586.20
21804	975.68	22703.2	69.76 (1.16)	23659.2	40629.8	71.73	83.4	100	1000	191.68	0	0	1000	191.60	383.28
41145.8	1896.20	41399	135.80 (8.64)	41399	41495.2	.23	150.2	203.2	1000	259.76	90.8	826.6	282.6	46.92	306.68

Tableau 54 (1000 variables)

Les résultats du tableau 54 montrent que pour certains problèmes de 1000 variables (i.e. séries 10 (m=10), 11 (m=10 et 100) et 12 (m=10 et 100)) le choix de

$$v(RK(\bar{\lambda}) \mid x_k = \varepsilon_k)$$

comme majorant de $v(K \mid x_k = \varepsilon_k)$ est insuffisant.

Des expériences additionnelles ont alors été réalisées en considérant deux autres types de relaxations de $(K \mid x_k = \varepsilon_k)$ explicités au § IV.5.2, utilisant

- les conditions de bornes sur $\sum_j x_j$, dans la relaxation lagrangienne
- la contraction de contraintes.

Le tableau 55 fournit un complément des résultats du tableau 54 ; il indique en particulier pour chacun des nouveaux choix de majorants (mis en oeuvre en séquence), les valeurs moyennes des suppléments de temps d'exécution nécessaires à l'élimination de nouvelles variables.

		Relaxation Lagrangienne			Contraction de contraintes			
b	m	$\bar{v}(K)$	$\|X_2\|$	Temps	$\bar{v}(K)$	$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	$\|X_2\|$	Temps
1000	10	3 175	174.8	31.04	2256	0.03	51.4*	33.92
10000	10	4161.4	281.8	65.56	2782	0.02	76.2*	84.28
	100	21665.4	419.8	88.73	23623.6	1.88	414.6*	140.5
50000	10	18417	1000	224.2	2963	0.004	811.8**	453.2
	100	30597	1000	218.6	28410.6	0.25	1000	978.32

* Pour 2 problèmes l'optimum est atteint

** $\ell = u = 10$ et $|X_1| = 4$

TABLEAU 55 (1000 VARIABLES)

L'analyse de ces résultats aboutit aux conclusions suivantes :

(i) pour n fixé

a/ variation avec m :

- La borne supérieure de la précision de notre méthode, ainsi que le nombre de variables non éliminées sont des fonctions décroissantes de m.

- L'encadrement de la somme des variables, la différence $u-\ell$, le temps de construction du convexe et celui de l'algorithme G_1 sont des fonctions croissantes de m.

- Par contre, les temps de l'algorithme G_2 sont peu sensibles à la variation de m (la croissance avec m des temps de constructions des convexes est compensée par la mise à un d'un plus grand nombre de variables à chaque itération) ; il en est de même pour le temps total de réduction, à l'exception des séries 5 et 7 pour lesquelles la croissance avec m est significative.

b/ variation avec b :

- L'élimination à 0 des variables, basée sur la comparaison des coefficients de la contrainte avec $b(\ell)$, est d'autant plus efficace que les valeurs prises par b sont petites.

- Le nombre de variables éliminées est une fonction croissante de l'amplitude des valeurs de b.

(ii) bornes inférieures de v(K) :

- L'algorithme G_2 est plus efficace que l'algorithme G_1 aussi bien en temps de calcul qu'en valeur.

- Les valeurs finales de $\underline{v}(K)$ sont en majorité obtenues par l'algorithme G_2 ; mais les algorithmes d'encadrement de $\sum_j x_j$, ainsi que les linéarisations majorantes ont une contribution non négligeable dans leurs déterminations.

(iii) variation des résultats avec n :

a/ pour m fixé et b prenant ses valeurs dans un intervalle donné : i.e. pour m = 10 et b(j) ϵ]0,1000[j=1,...,m - unique possibilité parmi l'éventail des problèmes tests - le tableau 56 montre en particulier la croissance linéaire du temps de calcul avec n.

b/ pour m $\simeq$ n/2 et b prenant ses valeurs dans un intervalle donné : (i.e. b(j) ϵ]0,1000[$\forall$ j) : les résultats du tableau 57 aboutissent à une conclusion identique quant à l'évolution du temps total. Par contre, la croissance de l'écart entre les temps des algorithmes G_1 et G_2 est ici beaucoup plus forte ; enfin, la borne supérieure de la précision ainsi que le nombre de variables non éliminées sont dorénavant des fonctions décroissantes de m, à l'instar du cas "n fixé".

n	Alg. G_1		Alg. G_2		$\frac{\bar{v}(K)-\underline{v}(K)}{\underline{v}(K)}$	ℓ	u	$\lvert X_2 \rvert$	Temps Total
	$\underline{v}(K)$	Temps	$\underline{v}(K)$	Temps					
50	1108.5	3.04	1129.5	3.06	7.61	4.6	6.1	6.1	13.76
100	1248.2	10.03	1261.8	7.06	11.30	4.8	6.5	20.8	24.92
1000	2008.2	78.21	2174.2	48.16	51.80	8	8.4	216.4	206.76

Tableau 56 (m = 10, b(j) ϵ]0,1000[$\forall$ j)

n	m	Alg. G_1		Alg. G_2		$\frac{\bar{v}(K)-v(K)}{\bar{v}(K)}$	ℓ	u	$\lvert X_2 \rvert$	Temps Total
		$\underline{v}(K)$	Temps	$\underline{v}(K)$	Temps					
50	20	1299.2	3.88	1342.2	3.80	5.43	5.6	7.4	7.1	16.74
	30	1389.3	5.14	1389.3	4.24	5.12	5.5	7.9	6.6	17.48
100	50	2178.2	13.16	2178.2	8.52	2.44	8.3	12.5	7.5	36.08
1000	500	7388.2	554	7384.4	87.60	.007	25.4	40	1.2	440.32

Tableau 57 ($m \simeq n/2$, $b(j) \in]0,1000[\ \forall j$)

Remarque : Comparaison des algorithmes G_1 et G_1' :

Une étude comparative empirique (tableaux 58 et 59) des algorithmes G_1 et G_1' - du point de vue temps de calcul puisqu'ils sont deux variantes d'une même méthode et aboutissent donc à des valeurs identiques - permet de conclure à la plus grande efficacité en moyenne de l'algorithme G_1'.

Il est à noter enfin que pour n = 100 et m = 10 et 30, les résultats sont à l'avantage de G_1 : l'efficacité de l'étape d'élimination de l'algorithme G_2 est d'autant moins grande que le nombre de variables mises à 1 est petit. □

b	m =	10		30		50		70		100	
		G_1	G_1'	G_1	G_1'	G_1	G_1'	G_1	G_1'	G_1	G_1'
5000	Ecart	11.4	10.6	29.0	19.2	32.0	21.2	32.0	23.6	38	26.6
	Maximum	9.0	11.6	16.6	19.6	23.2	25.0	22.2	23.8	-	-
	Plus mauvais cas	11.4	11.8	29.0	19.6	32	25	32	27.8	39	27.6
	Moyenne	9.95	10.68	18.78	18.58	22.2	21.36	24.22	24.24	32.56	22.23
10000	Ecart	20.8	11.6	20.6	19.2	31.0	24.4	37.0	26.6	40.2	32.2
	Maximum	8.8	10.6	20.8	22.8	22.8	24.2	27.2	29.0	29.0	30.2
	Plus mauvais cas	20.8	11.6	20.8	22.8	31.0	25.4	37.0	30.4	40.2	32.2
	Moyenne	10.54	10.94	19.42	19.82	24.74	24.12	29.34	28.66	31.42	30.78

Tableau 58 (n = 100)

SERIE	10		11		12	
Problème	G_1	G'_1	G_1	G'_1	G_1	G'_1
1	772.8	455.2	1335.6	989.8	1832.4	1852.4
2	447.6	545.2	1449.4	1139.6	2376.4	1921.8
3	640.4	427.2	926.6	929.4	1510.2	1744.6
4	447.6	621.4	1113.2	1100.0	1901.8	1909.6
5	461.6	458.8	1048.6	1014.8	1860.2	1777.2
Moyenne	554.0	501.52	1184.68	1034.72	1896.2	1841.12

Tableau 59 (n = 1000, m = 500)

A N N E X E

DONNÉES DES PROBLÈMES TESTS DE [EK9]

PROBLEME 1

c = 75 27 9 38 17 20 64 35 86 27 56 32 21 22 37
45 8 38 17 58 93 7 95 8 28 77 9 2 16 19

a = 31 96 16 64 52 46 7 13 42 48 4 6 83 65 96
14 46 44 45 33 60 77 98 3 31 65 6 94 66 83

b = 264 223 216 214 126 108 49 29 27 22 (m = 10)

$x_1^* = x_7^* = x_9^* = x_{11}^* = x_{12}^* = x_{16}^* = 1$; $ax^* = 104$; $cx^* = 358$

PROBLEME 2

c et a identiques aux données du problème 1

b = 290 269 261 252 249 234 230 229 221 207 188 176 172 165 159
158 157 155 150 120 117 111 91 82 77 48 20 11 9 2

$x_1^* = x_7^* = x_8^* = x_9^* = x_{11}^* = x_{12}^* = x_{16}^* = x_{20}^* = x_{21}^* = 1$; $ax^* = 210$; $cx^* = 544$

PROBLEME 3

c et a identiques aux données du problème 1

b(j) = 300 j = 1,...,30

$x_1^* = x_3^* = x_7^* = x_8^* = x_9^* = x_{11}^* = x_{12}^* = x_{16}^* = x_{20}^* = x_{21}^* = x_{24}^* = x_{26}^* = x_{27}^* = 1$;
$ax^* = 300$; $cx^* = 647$

PROBLEME 4

c = 276 71 111 234 233 272 180 15 16 178 212 89 207 65 62
165 204 229 279 228 5 211 51 240 167 110 226 47 230 4
201 288 241 209 98 81 55 80 287 87 96 245 104 199 175
77 225 61 120 112

a = 403 850 820 794 375 10 480 640 647 645 800 378 358 390 335
694 145 292 667 855 315 603 6 921 382 517 345 265 291 834
125 635 49 459 377 156 756 666 12 795 561 289 416 321 515
720 843 232 466 374

b = 978 862 712 470 445 437 379 233 201 35 (m = 10)

$x_6^* = x_{17}^* = x_{23}^* = x_{31}^* = x_{33}^* = x_{39}^* = 1$; $ax^* = 347$; $cx^* = 1256$

PROBLEME 5

c = 249 298 151 274 248 29 64 75 165 290 121 140 269 131 175
224 286 106 97 82 130 201 52 174 87 190 272 245 287 61
211 255 250 225 74 187 251 138 143 126 144 221 185 43 78
108 81 170 238 4 115 300 36 240 154 261 104 294 256 112
232 210 7 193 246 91 92 204 244 11 148 38 6 169 253
280 42 23 168 53 68 291 161 73 191 226 275 282 270 277
85 119 216 65 129 34 179 228 197 258

a = 506 231 66 748 638 230 498 249 326 316 241 954 761 769 432
343 378 618 762 752 341 817 251 821 930 330 853 50 502 368
84 952 394 269 487 914 539 484 220 899 535 22 510 643 530
246 759 790 598 58 34 708 882 830 449 794 456 844 150 595
542 632 967 397 120 227 153 281 570 61 367 593 428 1000 555
708 848 534 966 655 824 721 182 434 304 125 692 264 672 563
837 492 697 186 461 903 664 303 519 733

b = 978 862 712 470 445 437 379 233 201 35 (m = 10)

$x_{28}^* = x_{31}^* = x_{42}^* = x_{51}^* = x_{65}^* = x_{86}^* = 1$; $ax^* = 435$; $cx^* = 1264$

BIBLIOGRAPHIE

Les articles et ouvrages sont classés en six catégories caractérisées par une (ou deux) lettre(s), regroupant les travaux :

		Pages
(G)	d'intérêt *Général* sur la programmation mathématique en nombres entiers	1
(C)	traitant de la *Complexité* des algorithmes	8
(A)	décrivant des méthodes *Approchantes* pour le problème du knapsack	11
(E)	relatifs à la contraction de contraintes (*Equivalence* entre un système de contraintes et une contrainte unique)	13

consacrés aux

(K)	méthodes exactes pour le problème du *knapsack*	15
(EK)	méthodes exactes pour les *Extensions* du problème du *knapsack*	20

Tout article ou ouvrage est référencé par un nombre précédé de la (ou les) lettre(s) caractérisant sa catégorie d'appartenance.

Travaux d'intérêt général sur la programmation mathématique en nombres entiers

[G1] BALAS E., "An Additive Algorithm for solving linear programs with zero-one variables", Operations Research 4 (1965) 517-546.

[G2] BALAS E., "Discrete programming by the filter method", Operations Research 15 (1967) 915-957.

[G3] BALAS E. et MARTIN C.H., "Pivot and Complement-A Heuristic for 0-1 Programming", Management Sciences Research Report n° 414 (1978).

[G4] BELL D.E., "A Simple Algorithm for Integer programs using group constraints", Operational Research Quarterly 28 (2) (1977) 453-458.

[G5] BELLMAN R., "Some applications of the theory of dynamic programming. A review", Operations Research 2 (1954) 275-288.

[G6] BRADLEY G.H., HAMMER P.L. et WOLSEY L., "Coefficient reduction for inequalities in 0-1 Variables", Mathematical Programming 7 (1974) 263-282.

[G7] BREU R. et BURDET C.A., "Branch and Bound Experiments in Zero-One Programming", Mathematical Programming Study 2 (1974) 1-50.

[G8] BURDET C.A. et JOHNSON E.L., "A subadditive approach to the group problem of integer programming", Mathematical Programming 2 (1974) 51-71.

[G9] CHASSERY J.M., "Convexité et enveloppe convexe dans Z x Z", Séminaire d'Analyse Numérique, rapport n° 297 de l'Université Scientifique et Médicale de Grenoble (1978).

[G10] CHRISTOFIDES N. et WHITLOCK C., "An algorithme for Two-Dimensional Cutting problems", Operations Research 25 (1) (1977) 30-44.

[G11] CORD J., "A method for allocating funds to investment projects when returns are subject to uncertainty", Management Science 10 (2) (1964) 335-341.

[G12] ETCHEBERRY J., CONCA C. et STACCHETI E., "An Implicit enumeration Approach for integer programming using subgradient optimization", Serie Documentos de Trabajo-Centro de Planeamiento, Universidad de Chile, n° 4 (1978).

[G13] FAYARD D. et PLATEAU G., "Résolution d'un problème d'affectation", Bulletin de la Direction des Etudes de Recherches E.D.F. série C(1) (1973) 83-108.

[G14] FAYARD D. et PLATEAU G., "Programmation linéaire bivalente : un algorithme efficace et applications", Cahiers du Centre d'Etudes de Recherche Opérationnelle 16 (4) (1974) 383-411.

[G15] FIOROT J.CH., "Structure d'ensembles de points entiers", Thèse de Doctorat de spécialité, Université des Sciences et Techniques de Lille 1 (1971).

[G16] FISCHER M.L., "Lagrangean relaxation methods for combinatorial optimization", Proceedings of the Summer School in combinatorial Optimization, Urbino (Italie) (1978).

[G17] FREHEL J., "Sur l'utilisation de troncatures de Gomory dans les algorithmes énumératifs", Revue française d'Automatique, Informatique et Recherche Opérationnelle R-2 (1973) 5-15.

[G18] GARFINKEL R.S. et NEMHAUSER G.L., "Integer Programming", Wiley-Inter-Science, Wiley J. (Ed), New-York, London, Sydney, Toronto (1972).

[G19] GENUYS F., "Application de la programmation linéaire en nombres entiers à un problème de découpe", Proceedings of the I.F.I.P. Congress (1962).

[G20] GEOFFRION A.M., "Integer programming by implicit enumeration and Balas'method", SIAM Review 9(2) (1967) 178-190.

[G21] GEOFFRION A.M., "An improved implicit enumeration approach for integer programming", Operations Research 17 (1969) 437-454.

[G22] GEOFFRION A.M., "Elements of large-scale mathematical programming part I : Concepts", Management Science 16 (11) (1970) 652-675.

[G23] GEOFFRION A.M., "Lagrangean relaxation for integer programming", Mathematical Programming Study 2 (1974) 82-114.

[G24] GILMORE P.C., "Cutting stock, linear programming, knapsacking, dynamic programming and integer programming, some interconnections", Research Report n° RC 6528 Mathematical Sciences Department IBM Thomas J. Watson, New-York (1977).

[G25] GLOVER F., "A multiphase-dual algorithm for the zero-one integer programming problem", Operations Research 13 (1965) 879-919.

[G26] GLOVER F., "Surrogate constraints", Operations Research 16 (4) (1968) 741-749.

[G27] GLOVER F., "Neglected Heuristics in Integer Programming", Workshop on Integer Programming, Bonn (1975).

[G28] GLOVER F. et KLINGMAN D., "Mathematical Programming Models and Methods for the journal Selection Problem", Management Science Report Series Report n° 71-10, Business Research Division, Graduate School of Business Administration, University of Colorado (1971).

[G29] GOMORY R.E., "An algorithm for integer solutions to linear programs", Recent Advances in Mathematical Programming, Graves et Wolfe (Ed), New-York, Mc Grawhill (1963).

[G30] GOMORY R.E., "On the relation between integer and non integer solutions to linear programs", Proceedings of the National Academy of Sciences 53 (1965) 260-265.

[G31] GONDRAN M., "Un outil pour la programmation en nombres entiers : "La méthode des congruences décroissantes"", Revue française d'Automatique, Informatique, Recherche Opérationnelle V-3 (1973) 35-54.

[G32] GONDRAN M., "*Programmation combinatoire*", Bulletin de la Direction des Etudes et Recherches E.D.F. Série C (2) (1974) 45-66.

[G33a] GUIGNARD M. et SPIELBERG K., "*Algorithms for exploiting the structure of the simple plant location problem*", Annals of Discrete Mathematics 1 (1977).

[G33b] GUIGNARD M. et SPIELBERG K., "*Propagation, Penalty Improvement and Use of Logical Inequalities in Interactive (0-1) Programming*", International Symposium on Mathematical Programming, Heidelberg (1976).

[G34] HAMMER P.L. et RUDEANU S., "*Boolean Methods in Operations Research and Related Areas*", Springer-Verlag (1968).

[G35] HAMMER P.L., "*Boolean Elements in Combinatorial Optimization:A Survey*", Department of Combinatorics and Optimization, University of Waterloo Canada (1977).

[G36] HANSEN P., "*Programmes Mathématiques en variables 0-1*", Thèse d'Agrétation, Faculté des Sciences Appliquées de l'Université Libre de Bruxelles (1974).

[G37] HANSSMANN F., "*Operations Research in the National planning of Underdeveloped countries*", Operations Research 9 (1961) 230-248.

[G38] HUARD P. et GUIGOU J., "*Programmation mathématique en variables bivalentes : Prise en compte d'équations linéaires*", Note E.D.F., n° HR 7261 (1967).

[G39] KAPLAN S., "*Solution of the Lorie-Savage and Similar Integer Programming Problems by the generalized Lagrange Multiplier Method*", Operations Research 14 (1966) 1130-1136.

[G40] KIANFAR F., "*Stronger inequalities for 0-1 integer programming using knapsack functions*", Operations Research 19 (5) (1971) 1374-1392.

[G41] KIANFAR F., "*Stronger inequalities for 0-1 integer programming : computational refinements*", Operations Research 24 (3) (1976) 581-585.

[G42] KRAFT D.M. et HILL T.W., "*A Lagrangian Formulation of the journal selection Model*", Technical Report, School of Industrial Engineering, Purdue University (1971).

[G43] LAND A.H. et DOIG A.G., "An automatic method of solving discrete programming problems", Econometrica 28 (3) (1960) 497-520.

[G44] LEGENDRE J.P. et MINOUX M., "Une application de la notion de dualité en programmation en nombres entiers : Selection et affectation optimales d'une flotte d'avions", Revue française d'Automatique, d'Informatique et de Recherche Opérationnelle 11 (2) (1977) 201-222.

[G45] LITTLE J., MURTY K., SWEENEY D. et KAREL C., "An Algorithm for the Traveling Salesman Problem", Operations Research 11 (1963) 972-989.

[G46] MEVERT P. et SUHL U., "Implicit Enumeration with generalized upper bounds", Annals of Discrete Mathematics 1 (1977) 393-402.

[G47] PADBERG M.W.,"Covering, packing and knapsack problems", Research Report RC 6658, IBM T.J. Watson Research Center (1977).

[G48] PETERSEN C.C., "Computational experience with variants of the Balas algorithm applied to the selection of R & D projects", Management Science 13 (9) (1967) 736-750.

[G49] PLATEAU G, "Réduction de la taille des problèmes linéaires en variables 0-1", Publication 71 du Laboratoire de Calcul de l'Université des Sciences et Techniques de Lille 1 (1976).

[G50] PREPARATA F.P., et HONG S.J., "Convex hulls of finite sets of points in two and three dimensions", Communications of the Association for Computing Machinery 20 (2) (1977) 87-93.

[G51] ROSS G.T. et SOLAND R.M., "A branch and bound algorithm for the general assignment problem", Mathematical Programming 8 (1975) 91-103.

[G52] ROY B., NGHIEM P.T. et BERTIER P., "Programmes linéaires en nombres entiers et procédure S.E.P.", Metra 4 (3) (1965) 441-460.

[G53] SALKIN H.M., *"Integer Programming"*, Addison-Wesley Publishing Company (1975).

[G54] SALKIN H.M., MORITO S. et MATHUR K., *"The knapsack Problem : Applications, Algorithms, and Computations"*, North-Holland Publishing Company, à paraître.

[G55] SAUNDERS R.M. et SCHINZINGER R., *"A shrinking boundary algorithm for discrete system models"*, I.E.E.E. Transactions on systems science and cybernetics SSC-6 (2) (1970) 133-140.

[G56] SHAPIRO J.F., *"Dynamic programming algorithms for the integer programming problem - I : the integer programming problem viewed as a knapsack type problem"*, Operations Research 16 (1968) 103-121.

[G57] SHIH W., *"A branch and Bound Method for the Multiconstraint zero-one knapsack problem"*, Journal of the operational Research Society 30 (4) (1979) 369-378.

[G58] SPIELBERG K.,*"Enumerative Methods in Integer Programming"*, Annals of Discrete Mathematics 5 (1979).

[G59] STEINMANN H. et SCHWINN R.,*"Computational experience with a zero-one programming problem"*, Operations Research 17 (1969) 917-920.

[G60] TOYODA Y., *"A Simplified Algorithm for obtaining Approximate Solutions to zero-one programming Problems"*, Management Science 21 (12) (1975) 1417-1427.

[G61] TRAUTH C.A. et WOOLSEY R.E., *"Integer linear programming : a study in computational efficiency"*, Management Science 15 (9) (1969) 481-493.

[G62] WALUKIEWICZ S. et KALISZEWSKI I., *"Tighter equivalent formulations of integer programming problems"*, présenté au Congrès "Methods of Mathematical Programming", Zakopane (Pologne) (1977).

[G63] WEINGARTNER H.M. et NESS D.N., "*Methods for the solution of the multidimensional 0/1 knapsack problem*", Operations Research 15 (1) (1967) 83-103.

[G64] ZIONTS S., "*Generalized implicit enumeration using bounds on variables for solving linear programs with zero-one variables*", Naval Research Logistic Quarterly 19 (1972) 165-181.

[G65] ZIONTS S., "*Linear and Integer Programming*", Prentice Hall (1974).

Complexité des algorithmes

[C1] AHO A.V., HOPCROFT J.E. et ULLMAN J.D., "*The design and analysis of computer algorithms*", Addison-Wesley Publishing Company (1976).

[C2] BLUM M., FLOYD R.W., PRATT V., RIVEST R.L. et TARJAN R.E., "*Time bounds for selection*", Journal of Computer and System Sciences 7 (1973) 448-461.

[C3] CHACIJAN L.G., "*Polynomial algorithm for linear programming*", présenté par Lawler E.L. au Colloque Franco-Canadien de Combinatoire (1979).

[C4] COOK S.A., "*The complexity of theorem proving procedures*", Proceedings 3rd. Ann. ACM Symposium Theory of Computing, ACM, N.Y., (1971) 151-158.

[C5] D'ATRI G. et PUECH C., "*Analyse probabilistique de problèmes combinatoires*", Rapport n° 3 du groupe de recherche 22 du C.N.R.S., Université de PARIS VI (1978).

[C6] FUSSENEGGER F. et GABOW H.N., "*A counting approach to lower bounds for selection problems*", Journal of the Association for Computing Machinery 26 (2) (1979) 227-238.

[C7] GAREY M.R. et JOHNSON D.S., "*Approximation algorithms for combinatorial problems : an annoted bibliography*", dans "Algorithms and complexity : New directions and recent results", TRAUB J.F. (Ed.), Academic Press, N.Y.,(1976) 41-52.

[C8] GAREY M.R. et JOHNSON D.S., "*Strong NP-Completeness results : motivation, examples and implications*", Journal of the Association for Computing Machinery 25 (3) (1978) 499-508.

[C9] HOARE C.A.R., "*Quicksort*", Computer Journal 5 (4) (1962) 10-15.

[C10] JEROSLOW R.G., "*Trivial integer programs unsolvable by branch-and-bound*", Mathematical Programming 6 (1974) 105-109.

[C11] KARP R.M., "*Reducibility among combinatorial problems*", complexity of computer computations de MILLER R.E., et THATCHER J.W., Plenum Press, New-York, (1972) 85-104.

[C12] KARP R.M., "*On the computational complexity of combinatorial problems*", Networks 5 (1975) 45-68.

[C13] KARP R.M., "*The probabilistic analysis of some combinatorial search algorithms*", dans "Algorithms and complexity : New directions and recent results", TRAUB J.F. (Ed.), Academic Press, N.Y.,(1976) 1-19.

[C14] KLEE V. et MINTY G.J., "*How good is the simplex algorithm ?*", dans Inequalities III, Shisha O. (Ed.), Academic Press, (1972) 159-175.

[C15] KNUTH D.E., "*The art of computer programming Vol. 3 : sorting and searching*", Addison-Wesley, Mass. (1972).

[C16] LENSTRA J.K. et RINNOOY KAN A.H.G., "*Computational complexity of discrete optimization problems*", Research Report B.W. 77/77, Department of Operations Research, Mathematisch Centrum, Amsterdam (Pays Bas) (1977).

[C17] MEYER B. et BAUDOIN C., "*Méthodes de programmation*", Eyrolles Paris (1978).

[C18] RABIN M.O., "*Probabilistic algorithms*", dans "Algorithms and complexity New directions and recent results", TRAUB J.F. (Ed.), Academic Press, N.Y., (1976) 21-40.

[C19] SAHNI S., "*General techniques for combinatorial approximation*", Operations Research 25 (6) (1977) 920-936.

[C20] SAHNI S. et HOROWITZ E., "*Combinatorial problems : reducibility and approximation*", Operations Research 26 (5) (1978) 718-759.

[C21] SEDGEWICK R., "*Implementing quicksort programs*", Communications of the Association for Computing Machinery 21 (10) (1978) 847-857.

[C22] TARJAN R.E., "*Complexity of combinatorial algorithms*", Computer Science Department, School of Humanities and Sciences, Stamford University (1977).

[C23] WEIDE B., "*Analysis techniques for discrete algorithms*", Computing Surveys 9 (4) (1977) 292-313.

Méthodes approchantes pour le problème du knapsack

[A1] CHANDRA A.K., HIRSCHBERG D.S. et WONG C.K., "*Approximate algorithms for the knapsack problem and its generalizations*", Research Report of the IBM Thomas J. WATSON Research Center, New-York (1975).

[A2] D'ATRI G., "*Probabilistic analysis of the knapsack problem*", Rapport n° 7 du groupe de recherche 22 du C.N.R.S., Université de PARIS VI (1978).

[A3] FISHER M.L., "*Worst-case analysis of heuristics*", Proceedings of the Summer School in Combinatorial Optimization, Urbino (Italie) (1978).

[A4] HIRSCHBERG D.S. et WONG C.K., "*A polynômial time algorithm for the knapsack problem with two variables*", Journal of the Association for Computing Machinery 23 (1) (1976) 147-154.

[A5] HU T.C. et LENARD M.L., "*Optimality of a heuristic solution for a class of knapsack problems*", Operations Research 24 (1) (1976) 193-196.

[A6] IBARRA O.H. et KIM C.E., "*Fast approximation algorithms for the knapsack and sum of subsets problems*", Journal of the Association for Computing Machinery 22 (4) (1975) 463-468.

[A7] JOHNSON D., "*Approximation algorithms for combinatorial problems*", Journal of Computer Systems Sciences 9 (1974) 256-278.

[A8] KANNAN R. et KORTE B., "*Approximative combinatorial algorithms*", Proceedings of the Summer School in Combinatorial Optimization, Urbino (Italie) (1978).

[A9] LAWLER E.L., "*Fast approximation algorithms for knapsack problems*", Research Report of the Electronics Research Laboratory, University of California, Berkeley (1977).

[A10] MAGAZINE M.J., NEMHAUSER G.L. et TROTTER L.E., *"When the greedy solution solves a class of knapsack problems"*, Operations Research 23 (2) (1975) 207-217.

[A11] MAGAZINE M.J. et OGUZ O., *"A fully polynomial approximate algorithm for the 0-1 knapsack problem"*, Working Paper 114 of the Department of Management Sciences, University of Waterloo, (1979).

[A12] SAHNI S., *"Approximate algorithm for the 0-1 knapsack problem"*, Journal of the Association for Computing Machinery 22 (1) (1975) 115-124.

Contraction de contraintes

[E1] ANTHONISSE J.M., "*A note on equivalent systems of linear diophantine equations*", Operations Research 17 (1973) 167-177.

[E2] BRADLEY G.H., "*Transformation of integer programs to knapsack problems*", Discrete Mathematics 1 (1971) 29-45.

[E3] CHVATAL V. et HAMMER P.L., "*Aggregation of inequalities in integer programming*", Annals of Discrete Mathematics 1 (1977) 145-162.

[E4] GLOVER F., "*New results on equivalent integer programming formulations*", Mathematical Programming 8 (1975) 84-90.

[E5] GLOVER F. et WOOLSEY R.E., "*Aggregating diophantine constraints*", Zeitschrift für Operations Research 16 (1972) 1-10.

[E6] KALISZEWSKI I. et LIBURA M., "*Constraints aggregation in integer programming*", Report MPD 5-77, Systems Research Institute, Polish Academy of Sciences, Varsovie (Pologne) (1977).

[E7] KENDALL K.E. et ZIONTS S., "*Solving integer programming problems by aggregating constraints*", Operations Research 25 (2) (1977) 346-351.

[E8] MATHEWS G.B., "*On the partition of numbers*", Proceedings of the London Mathematical Society 28 (1896) 486-490.

[E9] MEYER R.R., "*Equivalent constraints for discrete sets*", Report 1748, Mathematics Research Center, University of Wisconsin, Madison (1977).

[E10] PADBERG M.W., "*Equivalent knapsack-type formulations of bounded integer linear progams. : an alternative approach*", Naval Research Logistic Quarterly 19 (1972) 699-708.

[E11] WEINBERG F., "*A necessary and sufficient condition for the aggregation of linear diophantine equations*", IFØR-Studienberichte, Heft 4 (1976).

[E12] WEINBERG F., "A *spectral algorithm for sequential aggregation of m linear diophantine constraints*", IFØR-Studienberichte, Heft 5 (1976).

Méthodes exactes pour le problème du knapsack

[K1] AHRENS J.H. et FINKE G., "*Merging and sorting applied to the zero-one knapsack problem*", Operations Research 23 (6) (1975) 1099-1109.

[K2] BALAS E. et ZEMEL E., "*Solving large zero-one knapsack problems*", Management Sciences Research Report (1977).

[K3] BARR R.S. et ROSS G.T., "*A linked list data structure for a binary knapsack algorithm*", Research Report of the Center for Cybernetic Studies, University of Texas, Austin (1975).

[K4] CABOT A.V., "*An enumeration algorithm for knapsack problems*", Operations Research 18 (2) (1970) 306-311.

[K5] DANTZIG G.B., "*Discrete-variable extremum problems*", Management Science 5 (1957) 266-277.

[K6] DEMBO R.S. et HAMMER P.L., "*A reduction algorithm for knapsack problems*", présenté au Congrès ORSA-TIMS, San Juan (1974).

[K7] DEMBO R.S. et HAMMER P.L., "*Preprocessing knapsack problems*", Research Report n° 26 of the School of Organization and Management, Yale University (1978).

[K8] FAYARD D. et PLATEAU G., "*Contribution à la résolution du problème du knapsack*", Thèse de Doctorat de Spécialité, Université des Sciences et Techniques de Lille 1 (1971).

[K9] FAYARD D. et PLATEAU G., "*Résolution du problème du knapsack en variables bivalentes ; comparaison de méthodes*", Bulletin de la Direction des Etudes et Recherches E.D.F. Serie C(1) (1972) 5-42.

[K10] FAYARD D. et PLATEAU G., "*Resolution of the 0-1 knapsack problem : comparison of methods*", Mathematical Programming 8 (3) (1975) 272-307.

[K11] FAYARD D. et PLATEAU G., "*Techniques de résolution du problème du knapsack en variables bivalentes*",
Partie 1, Bulletin de la Direction des Etudes et Recherches E.D.F., série C (1) (1976) 51-75.

[K12] Partie 2, Publication 84 du Laboratoire de Calcul de l'Université des Sciences et Techniques de Lille 1 (1977).

[K13] Partie 3, Publication 91 du Laboratoire de Calcul de l'Université des Sciences et Techniques de Lille 1 (1977).

[K14] FAYARD D. et PLATEAU G., " *Reduction algorithms for single and multiple constraints 0-1 linear programming problems*", Publication 92 du Laboratoire de Calcul de l'Université des Sciences et Techniques de Lille 1, et Proceedings du Congrès "Methods of Mathematical Programming", Zakopane (Pologne) (1977).

[K15] FAYARD D. et PLATEAU G., "*On "an efficient algorithm for the 0-1 knapsack problem, by Robert M. NAUSS""*, Management Science 24 (9) (1978) 918-919.

[K16] FAYARD D. et PLATEAU G, "*Le problème du knapsack en variables bivalentes : relaxation, réduction, applications*", Actes du Congrès A.F.C.E.T. - S.M.F., Palaiseau (1978).

[K17] GILMORE P.C. et GOMORY R.E., "*A linear programming approach to the cutting-stock problem*", Operations Research 9 (1961) 849-859.

[K18] GILMORE P.C. et GOMORY R.E., "*A linear programming approach to the cutting-stock problem*", Operations Research 11 (1963) 863-888.

[K19] GILMORE P.C. et GOMORY R.E., "*Multistage cutting-stock problem of two and more dimensions*", Operations Research 13 (1965) 94-120.

[K20] GILMORE P.C. et GOMORY R.E., "*The theory and computation of knapsack functions*", Operations Research 14 (1966) 1045-1074.

[K21] GONDRAN M., "*Résolution du problème du sac à dos en utilisant la méthode des congruences décroissantes*", Note E.D.F. HI 1020/02 (1972).

[K22] GREENBERG H., "*An algorithm for the computation of knapsack functions*", Journal of Mathematical Analysis and Applications 26 (1969) 159-162.

[K23] GREENBERG H. et HEGERICH R.L., "A branch search algorithm for the knapsack problem", Management Science 16 (5) (1970) 327-332.

[K24] GUIGNARD M. et SPIELBERG K., "Mixed-integer algoirthms for the (0-1) knapsack problem", IBM Journal of Research and Development 16 (4) (1972) 424-430.

[K25] HAMMER P.L., PADBERG M.W. et PELED U.N., "Constraint pairing in integer programming", Department of combinatorics and optimization Research Report CORR 73-7 (1973).

[K26] HANSEN P., "Programmes mathématiques en variables 0-1", Thèse d'agrégation, Faculté des Sciences Appliquées de l'Université Libre de Bruxelles (1974).

[K27] HOROWITZ E. et SAHNI S., "Computing partitions with applications to the knapsack problem", Journal of the Association for Computing Machinery 21 (2) (1974) 277-292.

[K28] INGARGIOLA G.P. et KORSH J.F., "Reduction algorithm for zero-one single knapsack problems", Management Science 20 (4) (1973) 460-463.

[K29] INGARGIOLA G.P. et KORSH J.F., "A general algorithm for one-dimensional knapsack problems", Operations Research 25 (5) (1977) 752-759.

[K30] KOLESAR P.J., "A branch and bound algorithm for the knapsack problem", Management Science 13 (9) (1967) 723-735.

[K31] LAURIERE M., "An algorithm for the 0-1 knapsack problem", Mathematical Programming 14 (1) (1978) 1-10.

[K32] MARTELLO S. et TOTH P., "Branch and bound algorithms for the solution of the general unidimensional knapsack problem", présenté au Congrès EURO II (1976).

[K33] MARTELLO S. et TOTH P., "A computational study on large-size unidimensional knapsack problems", présenté au Congrès TIMS/ORSA, San Franscisco (1977).

[K34] MARTELLO S. et TOTH P., "*An upper bound for the zero-one knapsack problem and a branch and bound algorithm*", European Journal of Operational Research 1 (1977) 169-175.

[K35] MARTELLO S. et TOTH P., "*Algorithm for the solution of the 0-1 single knapsack*", Computing 21 (1978) 81-86.

[K36] MULLER-MERBACH H., "*An improved upper bound for the zero-one knapsack problem : a note on the paper by Martello et Toth*", European Journal of Operational Research 2 (1978) 212-213.

[K37] NAUSS R.M., "*An efficient algorithm for the 0-1 knapsack problem*", Management Science 23 (1) (1976) 27-31.

[K38] PLATEAU G., "*Le problème du knapsack en variables bivalentes : méthodes approximantes, élimination de variables*", Publication n° 54 du Laboratoire de Calcul de l'Université des Sciences et Techniques de Lille 1 (1975).

[K39] SALKIN H.M. et DE KLUYVER C.A., "*The knapsack problem : a survey*", Naval Research Logistics Quarterly 22 (1) (1975) 127-144.

[K40] SOSINSKI R., "*New tools for 0-1 knapsack problem - O(n) reduction algorithm and dominance constraints*", Article présenté au 9ème Symposium International sur la Programmation Mathématique, Budapest (1976).

[K41] SUHL U., "*An algorithm and efficient data structures for the binary knapsack problem*", European Journal of Operational Research 2 (1978) 420-428.

[K42] TOTH P., "*A new reduction algorithm for 0-1 knapsack problems*", présenté au Congrès ORSA-TIMS, Miami (Floride) (1976).

[K43] TOTH P., "*A dynamic programming algorithm for the zero-one knapsack problem*", présenté au Congrès TIMS/ORSA, San Francisco (1977).

[K44] WALUKIEWICZ S., "*The size reduction of a binary knapsack problem*", Bulletin de l'Académie des Sciences Polonaise, Série Sciences et Techniques 23 (5) (1975).

[K45] ZOLTNERS A.A., "*A direct descent binary knapsack problem*", Journal of the Association for Computing Machinery 25 (2) (1978) 304-311.

Méthodes exactes pour les extensions du problème du knapsack

[EK1] EILON S. et CHRISTOFIDES N., "*The Loading Problem*", Management Science 17 (1971) 259-268.

[EK2] FAYARD D. et PLATEAU G., "*Le Problème du knapsack "baudruche" en variables 0-1*", Publication ANO-1, Laboratoire de Calcul, Université des Sciences et Techniques de Lille 1 (1979).

[EK3] GALLO G., HAMMER P.L. et SIMEONE B., "*Quadratic knapsack problems*", Research Report 76-43, Department of Combinatorics and Optimization, University of Waterloo (Canada) (1976).

[EK4] INGARGIOLA G. et KORSH J.F., "*An Algorithm for the Solution of 0-1 Loading Problems*", Operations Research 23 (6) (1975) 1110-1119.

[EK5] ISHII H., IBARAKI T. et MINE H., "*Fractional knapsack problems*", Mathematical Programming 13 (1977) 255-271.

[EK6] MARTELLO S. et TOTH P., "*Solution of the zero-one multiple knapsack problem*", presented at the ORSA/TIMS, Atlanta (1977).

[EK7] MORIN T.L. et MARSTEN R.E., "*An algorithm for nonlinear knapsack problems*", Management Science 22(10) (1976) 1147-1158.

[EK8] NAUSS R.M., "*The 0-1 knapsack problem with multiple choice constraints*", European Journal of Operational Research 2 (1978) 125-131.

[EK9] POSNER M.E. et GUIGNARD M., "*The collapsing 0-1 knapsack problem*", Mathematical Programming 15 (2) (1978) 155-161.

[EK10] SUZUKI H., "*A generalized knapsack problem with variable coefficients*", Mathematical Programming 15 (1978) 162-176.

CODE DE

L'ALGORITHME MSB 79

```
0002            SUBROUTINE MSB79  (C,A,B ,N,XG,N1,ISORT)
                C
                C       THIS SUBROUTINE SOLVES THE 0-1 KNAPSACK PROBLEM
                C          ...  N1=C*XG=MAXIMIZE    C(1)*X(1)+...+C(N)*X(N)
                C           .                   SUBJECT TO
                C       (B).
                C           .                          A(1)*X(1)+...+A(N)*X(N)<=B
                C          ...                         X(J)=0 OR 1       J=1,....,N
                C       ALL PARAMETERS ARE INTEGER
                C
                C       INPUT PARAMETERS :
                C         - VECTORS C AND A     ....
                C                                . INCLUDED IN (B)
                C         - SCALARS B  AND N    ....
                C         - SCALAR ISORT DEALS WITH  THE SORTING OF THE DATA
                C                        OF THE REDUCED PROBLEM BY DECREASING
                C                        ORDER OF THE REDUCED COSTS IN ABSOLUTE VALUE
                C                         .... 0 IF QUICKSORT ALGORITHM  IS USED
                C                  ISORT =
                C                          ....  THRESHOLD VALUE UNDER WHICH THE
                C                                SUBFILES PRODUCED DURING SORTING ARE IGNORED
                C                                      N < 1000        ISORT =  0
                C                                      N = 1000        ISORT =  8
                C                                      N = 2000        ISORT = 10
                C                                      N = 5000        ISORT = 35
                C                                      N = 7500        ISORT = 60
                C
                C       OUTPUT PARAMETERS :
                C         - SCALAR N1 = OPTIMAL VALUE     ....
                C                                         . OF (B)
                C         - VECTOR XG = OPTIMAL SOLUTION ....
                C
                C
0003            INTEGER A(N),C(N),XG(N)
0004            DIMENSION IP(7500),DA(7501),IAUX1(100),IAUX2(100)
0005            INTEGER XE(7500),X(7500)
0006            INTEGER SUP,DIF,CM,CM2,SUP2,VE,EV,XM,XM1,XM2,PLUS
0007            INTEGER EX,XGM,S,B
0008            REAL K2,K2R,N3K2,IVAL
0009            DO 2 I=1,N
0010          2 IP(I)=I
0011            L1=B
                C
0012            CALL KLOK
                C
                C          SOLVING OF RELAXATION OF (B)
0013            S=0
0014            ISEUIL=10
0015            K=0
0016            DO 10 I=1,N
0017            R=A(I)
0018             X(I)=C(I)
0019           10 DA(I)=C(I)/R
```

```
0022          12 IB=IB1
0023             IH=IC1
0024             IF(IH-IB.GT.ISEUIL) GOTO 5
0026             J1=IB+1
0027          23 IF(J1.GT.IH) GOTO 301
0029             J2=J1-1
0030          91 IF(J2.LT.IB) GOTO 92
0032             IF(DA(J2).GE.DA(J2+1)) GOTO 92
0034             J3=J2+1
0035             IPV=IP(J2)
0036             VAL=DA(J2)
0037             IVAL=A(J2)
0038             IP(J2)=IP(J3)
0039             A(J2)=A(J3)
0040             DA(J2)=DA(J3)
0041             A(J3)=IVAL
0042             IP(J3)=IPV
0043             DA(J3)=VAL
0044             J2=J2-1
0045             GOTO 91
0046          92 J1=J1+1
0047             GOTO 23
0048           5 J1=IB
0049             J4=(IH+IB)/2
0050             J2=J4
0051             IF(DA(J1).LE.DA(J2)) GOTO 95
0053             J1=J4
0054             J2=IB
0055          95 J3=IH
0056             IF(DA(J3).GE.DA(J2)) GOTO 96
0058             J2=IH
0059             IF(DA(J3).GE.DA(J2)) J2=J1
0061          96 VAL=DA(J2)
0062             IVAL=A(J2)
0063             IPV=IP(J2)
0064             DA(J2)=DA(IH)
0065             A(J2)=A(IH)
0066             IP(J2)=IP(IH)
0067             IF(IB.GE.IH) GOTO 8
0069           6 IF (DA(IB).LT. VAL) GOTO 7
0071             K=K+A(IB)
0072             IB=IB+1
0073             IF(IB.GE.IH) GOTO 8
0075             GOTO 6
0076           7 DA(IH)=DA(IB)
0077             A(IH)=A(IB)
0078             IP(IH)=IP(IB)
0079           4 IH=IH-1
0080             IF(IB.GE.IH) GOTO 8
0082             IF(DA(IH).LE.VAL) GOTO 4
0084             DA(IB)=DA(IH)
0085             A(IB)=A(IH)
0086             IP(IB)=IP(IH)
0087             K=K+A(IH)
0088             IB=IB+1
0089             IF(IB.LT.IH) GOTO 6
```

```
0091          8 DA(IB)=VAL
0092            A(IB)=IVAL
0093            K=K+IVAL
0094            IP(IB)=IPV
0095            IF(K.LT.L1) GOTO 11
0097            K=K-A(IB)
0098            IF(K.LT.L1) GOTO 200
0100            IB=IB-1
0101            IF(K.EQ.L1) GOTO 199
0103            K=S
0104            IC1=IB
0105            GOTO 12
0106         11 IB1=IB+1
0107            S=K
0108            GOTO 12
0109        199 IB=IB-1
0110        200 IC=0
0111            DO 152 I=1,N
0112            K13=IP(I)
0113        152 C(I)=X(K13)
0114            S=IB-1
0115            DO 99 I=1,S
0116         99 IC=IC+C(I)
0117            R=A(IB)
0118            R=IC+((L1-K)*C(IB))/R
0119            GOTO 307
0120        301 S=IB-1
0121            DO 151 I=1,N
0122            K13=IP(I)
0123        151 C(I)=X(K13)
0124            IC=0
0125            DO 303 I=1,S
0126        303 IC=IC+C(I)
0127            DO 304 I=IB,IH
0128            J=I
0129            K=K+A(I)
0130            IC=IC+C(I)
0131            IF(K.GE.L1) GOTO 305
0133        304 CONTINUE
0134        305 IF(K.EQ.L1) GOTO 306
0136            IB=J
0137            R=A(IB)
0138            K=K-A(IB)
0139            IC=IC-C(IB)
0140            R=IC+((L1-K)*C(IB))/R
0141            GOTO 307
0142        306 R=IC
0143            IB=J
0144        307 VE=L1-K
0145            IV2=VE
0146            N1=IC
0147            M=N
C
0148          CALL KLOK
C
C           GREEDY ALGORITHM FOR A LOWER BOUND OF THE VALUE OF (N)
```

```
0155            IF (VE-A(I)) GOTO 13
0156            NI=NI+C(I)
0157            VE=VE-A(I)
0158            IP(I)=IP(I)+100000
0159         13 IP(I)=-IP(I)
0160            IP(IB)=-IP(IB)
0161            IZ41=R
0162            FVC=R-N1-1
0163            R=A(IB)
0164            R=C(IB)/R
0165            M1=N+1
0166            M2=N-1
0167            M3=N-2
0168            DA(M1)=R
0169            IF(IZ41.EQ.N1) GOTO 188
                C          REDUCTION ALGORITHM
0171            I=1
0172            SUP=1
0173         16 DA(I)=ABS(C(I)-R*A(I))
0174            IF(DA(I).GT.FVC) GOTO 15
0176         17 DA(M)=ABS(C(M)-R*A(M))
0177            IF(DA(M).GT.FVC) GOTO 14
0179            M=M-1
0180            IF(M.NE.I) GOTO 17
0182            SUP=I
0183            GOTO 18
0184         14 R1=DA(M)
0185            DA(M)=DA(I)
0186            DA(I)=R1
0187            H=IP(M)
0188            IP(M)=IP(I)
0189            IP(I)=H
0190            H=A(M)
0191            A(M)=A(I)
0192            A(I)=H
0193            H=C(M)
0194            C(M)=C(I)
0195            C(I)=H
0196            M=M-1
0197         15 IF(I.GE.M)GOTO 19
0199            I=I+1
0200            GOTO 16
0201         19 SUP=I+1
0202         18 IF (SUP.LT.N) GOTO 88
0204        188 IV2=-1
0205            SUP=M1
0206            GOTO 93
0207         88 ISEUIL=8
0208            IF(ISORT.NE.0) ISEUIL=ISORT
                C          SORTING OF THE DATA OF THE REDUCED PROBLEM
0210            J=1
0211            IBB=SUP
0212            IBH=N
0213        202 IB=IBB
0214            IH=IBH
0215            IF(IH-IB.GT.ISEUIL) GOTO 203
```

```
0217            IF(ISORT.NE.0) GOTO 207
0219            J1=IB+1
0220         20 IF(J1.GT.IH) GOTO 207
0222            J2=J1-1
0223         21 IF(J2.LT.IB) GOTO 22
0225            IF(DA(J2).GE.DA(J2+1)) GOTO 22
0227            J3=J2+1
0228            IPC=C(J2)
0229            IPB=A(J2)
0230            IPV=IP(J2)
0231            IVAL=DA(J2)
0232            IP(J2)=IP(J3)
0233            DA(J2)=DA(J3)
0234            A(J2)=A(J3)
0235            C(J2)=C(J3)
0236            A(J3)=IPB
0237            C(J3)=IPC
0238            DA(J3)=IVAL
0239            IP(J3)=IPV
0240            J2=J2-1
0241            GOTO 21
0242         22 J1=J1+1
0243            GOTO 20
0244        203 J1=IB
0245            J4=(IH+IB)/2
0246            J2=J4
0247            IF(DA(J1).LE.DA(J2)) GOTO 35
0249            J1=J4
0250            J2=IB
0251         35 J3=IH
0252            IF(DA(J3).GE.DA(J2)) GOTO 36
0254            J2=IH
0255            IF(DA(J3).LT.DA(J1)) J2=J1
0257         36 IVAL=DA(J2)
0258            IPV=IP(J2)
0259            IPB=A(J2)
0260            IPC=C(J2)
0261            DA(J2)=DA(IBB)
0262            IP(J2)=IP(IBB)
0263            A(J2)=A(IBB)
0264            C(J2)=C(IBB)
0265            IF(IB.GE.IH) GOTO 210
0267        206 IF(DA(IH).GT.IVAL) GOTO 205
0269            IH=IH-1
0270            IF(IB.GE.IH) GOTO 210
0272            GOTO 206
0273        205 DA(IB)=DA(IH)
0274            IP(IB)=IP(IH)
0275            A(IB)=A(IH)
0276            C(IB)=C(IH)
0277        204 IB=IB+1
0278            IF(IB.GE.IH) GOTO 210
0280            IF(DA(IB).GE.IVAL) GOTO 204
0282            DA(IH)=DA(IB)
0283             IP(IH)=IP(IB)
0284              C(IH)=C(IB)
0285               A(IH)=A(IB)
```

```
      IF(IB.LT.IM) GOTO 206
  210 DA(IB)=IVAL
      IP(IB)=IPV
      A(IB)=IPB
      C(IB)=IPC
      I=IB
      IF (J.GT.100) GOTO 201
      IAUX1(J)=I
      IAUX2(J)=IBH
      J=J+1
      IF(I.LE.IBB+1) GOTO 207
      IBH=I-1
      GOTO 202
  207 J=J-1
      IF(J.EQ.0) GOTO 208
      I=IAUX1(J)+1
      IBH=IAUX2(J)
      IF(IBH.LE.I) GOTO 207
      IBB=I
      GOTO 202
  208 CONTINUE
      IF(SUP.EQ.1) GOTO 73
   93 ISU2=SUP-1
      DO 74 I=1,ISU2
      XE(I)=1
      IF(IP(I).GT.0) GOTO 74
      XE(I)=0
      IP(I)=-IP(I)
      IF(IP(I).LE.100000) GOTC 74
      XE(I)=1
      IP(I)=IP(I)-100000
   74 CONTINUE
      IF(IV2.LT.0) GOTO 70
   73 DO 75 I=SUP,N
      X(I)=0
      IF(IP(I).GT.0) GOTO 76
      XE(I)=0
      IP(I)=-IP(I)
      XG(I)=0
      IF(IP(I).LE.100000) GOTO 75
      IP(I)=IP(I)-100000
      XE(I)=1
      GOTO 75
   76 XE(I)=1
      XG(I)=1
   75 CONTINUE
C         PARAMETERS FOR THE RESOLUTION OF
C         THE AUXILIARY PROBLEM (DENOTED BY (P.A.) )
      LM=A(N)
      LM2=A(M2)
      CM=C(N)
      CM2=C(M2)
      EX=XG(M2)
      XGM=XG(N)
      IF(LM-LM2.LE.0) GOTO 81
      SUP2=LM
      INF=LM2
      J=N
```

```
0355            GOTO 82
0356         81 SUP2=LM2
0357            INF=LM
0358            J=N-1
0359         82 IZ=0
0360            IF(C(J).LT.C(2*N-1-J)) IZ=1
0362            LS=LM+LM2
0363            DIF=EV+LM2*EX+LM*XGM
0364            M=N
0365            N3=IC
0366            N=N3-CM2*EX-CM*XGM
0367            N3K2=N
0368            K=SUP
          C
          C        IMPLICIT ENUMERATION ALGORITHM
          C        NEW CURRENT SOLUTION BY SOLVING (P.A.)
          C
0369            CALL KLOK
          C
0370            GOTO 86
0371         56 X(K)=1
0372            TP2=N
0373            DIF1=DIF
0374            IF(XG(K).NE.1)GOTO 27
0376            N=N-C(K)
0377            DIF=DIF+A(K)
0378            GOTO 28
0379         27 N=N+C(K)
0380            DIF=DIF-A(K)
0381         28 N=N-PLUS
0382            IF(DIF.GE.0) GOTO 86
0384            IF(K.NE.M3)GOTO 58
0386            X(K)=0
0387            IF(XG(K).EQ.0)K2R=K2R+A(K)
0389            DIF=DIF1
0390            N=TP2
0391            GOTO 48
0392         86 IF(DIF.GE.LS) GOTO 66
0394            IF(DIF.LT.INF) GOTO 65
0396            IF(IZ.EQ.1) GOTO 61
0398            IF(DIF.GE.SUP2) GOTO 64
0400         61 IF(INF.EQ.LM) GOTO 63
0402         62 XM2=1
0403            XM=0
0404            XM1=DIF-LM2
0405            GOTO 67
0406         63 XM2=0
0407            XM=1
0408            XM1=DIF-LM
0409            GOTO 67
0410         64 IF(INF.EQ.LM) GOTO 62
0412            GOTO 63
0413         65 XM2=0
0414            XM=0
0415            XM1=DIF
0416            GOTO 67
0417         66 XM2=1
```

```
0422              N=N+PLUS
0423              N3=N
0424              IF(N.LE.N1)GOTO 47
                C     NEW CURRENT OPTIMAL SOLUTION
0426              FVC=FVC+N1-N
0427              N1=N
0428              IF(SUP.GT.M3)GOTO 43
0430              DO 42 I=SUP,M3
0431              XE(I)=XG(I)
0432              IF(X(I).EQ.1) XE(I)=1-XG(I)
0434           42 CONTINUE
0435           43 XE(M2)=XM2
0436              XE(M)=XM
0437              VE=XM1
0438              IF(IZ41.EQ.N1) GOTO 70
0440              IF(SUP .GT.M1)GOTO 45
0442              ISU2=SUP
0443              DO 44 I=ISU2,M1
0444              IF(DA(I).LT.FVC)GOTO 45
0446              SUP=SUP+1
0447              IF(X(I).EQ.1)GOTO 70
0449           44 CONTINUE
0450           45 IF(SUP .GE.M)GOTO 70
0452              IF(ISORT.NE.0) GOTO 47
0454           46 IF(X(SUP).NE.1) GOTO 47
0456              SUP=SUP+1
0457              IF(SUP.GE.M)GOTO 70
0459              GOTO 46
                C     IMPLICIT LEXICOGRAPHICAL SEARCH
0460           47 K=M2
0461              Z=DA(M1)*XM1
0462              K2R=DIF
0463              IF(XM2.NE.EX)Z=Z+DA(M2)
0465              IF(XM.NE.XGM) Z=Z+DA(M)
0467           48 IF(K.EQ.SUP)GOTO70
0469              K=K-1
0470              IF(X(K).NE.1)GOTO 50
0472              X(K)=0
0473              IF(XG(K).NE.1)GOTO 49
0475              DIF=DIF-A(K)
0476              N=N+C(K)
0477              GOTO 51
0478           49 DIF=DIF+A(K)
0479              N=N-C(K)
0480              K2R=K2R+A(K)
0481           51 Z=Z+DA(K)
0482              GOTO 48
                C     STEP 6
0483           50 K2=-DA(K)+Z
0484              IF(N3+K2.LE.N1)GOTO 57
0486              IF(XG(K).NE.1)GOTO 52
0488              K2R=K2R+A(K)
0489              GOTO 53
0490           52 K2R=K2R-A(K)
                C     STEP 1 AND STEP 2
```

```
0491          53 CONTINUE
0492             IF(K2R.GE.0)GOTO 54
0494             IF(XG(K).EQ.0)  K2R=K2R+A(K)
0496             GOTO 48
              C        STEP 3,5 AND 6
0497          54 EV=K2R
0498             IS2=K+1
0499             IF(IS2.GT.M) GOTO 77
0501             DO 55 I=IS2,M
0502             IF(A(I).LE.K2R)EV=EV-A(I)
0504          55 CONTINUE
0505          77 IF(EV.LT.0)GOTO 56
0507             IF(N3+K2-DA(M1)*EV.GT.N1)GOTO 83
0509             IF(XG(K).EQ.0)K2R=K2R+A(K)
0511             GOTO 48
0512          57 IF(XG(K).EQ.1) K2R=K2R+A(K)
0514             GOTO 48
0515          83 IF(IS2.GT.M3) GOTO 56
0517             DO 84 I=IS2,M3
0518             IF(A(I).GT.K2R) GOTO 87
0520             IF(XG(I).EQ.1) GOTO 84
0522             X(I)=1
0523             DIF=DIF-A(I)
0524             N=N+C(I)
0525             GOTO 84
0526          87 IF(XG(I).EQ.0) GOTO 84
0528             X(I)=1
0529             DIF=DIF+A(I)
0530             N=N-C(I)
0531          84 CONTINUE
0532             GOTO 56
              C        EXPLICIT LEXICOGRAPHICAL SEARCH
0533          58 H=2
0534          59 J=M-H
0535             IF(X(J).EQ.1) GOTO 60
0537             IF(XG(J).EQ.0) GOTO 69
0539             X(J)=1
0540             DIF=DIF+A(J)
0541             N=N-C(J)
0542             IF(DIF.GE.0) GOTO 86
0544             GOTO 58
0545          60 X(J)=0
0546             N=N+C(J)
0547             DIF=DIF-A(J)
0548          69 H=H+1
0549             IF(J.GT.K+1)GOTO 59
0551             IF(K.EQ.SUP)GOTO 70
0553             X(K)=0
0554             IF(XG(K).EQ.0)K2R=K2R+A(K
0556             N=TP2
0557             DIF=DIF1
0558             GOTO 48
0559          70 CONTINUE
              C
0560             CALL KLOK
              C
              C        ALL OF THE RESULTS ARE GIVEN IN
                       [illegible] DATA [illegible] NO
```

```
0561            N=M
0562            IF(IV2) 72,89,72
0563         89 DO 173 I=1,M
0564        173 XE(I)=1
0565         72 CONTINUE
0566            DO 71 I=1,M
0567            K=IP(I)
0568            X(K)=C(I)
0569         71 XG(K)=XE(I)
0570            DO 104 I=1,M
0571            K=IP(I)
0572        104 XE(K)=A(I)
0573            DO 103 I=1,M
0574            C(I)=X(I)
0575        103 A(I)=XE(I)
0576            RETURN
0577        201 WRITE(2,102)
0578        102 FORMAT('  ARRAY OUT OF BOUNDS IN QUICKSORT SUBROUTINE ')
0579            RETURN
0580            END
```