REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère De L'enseignement Supérieur Et De La Recherche Scientifique
Université De Mouloud MAMMERI Tizi-Ouzou

Faculté Des Sciences
Département Mathématiques

Mémoire de fin d'étude

EN VUE DE L'OBTENTION DU DIPLOME DE MASTER EN RECHERCHE OPERATIONNELLE

Thème

Programmation Linéaire Mixte en Nombres Entiers

Présenté par:
M^{elle} AMANI Sabrina

Devant le jury:

M. AIDENE Mohamed	**Professeur**	**UMMTO**	**Président**
M. OUANES Mohand	**Professeur**	**UMMTO**	**Rapporteur**
M. CHEBBAH Mohamed	**M. de Conférences B**	**UMMTO**	**Examinateur**

Promotion : 2020/2021

Remerciements

Avant tout, je tiens à remercier Dieu le tout puissant pour m'avoir donnée la force, la volonté, la patience et le courage nécessaire pour réaliser ce travail.

Je souhaite tout particulièrement exprimer ma plus profonde reconnaissance à mon promoteur Monsieur OUANES de m'avoir encadrée, orientée, guidée et conseillée, mais également pour son aide précieuse, sa sympathie, sa gentillesse, et sa disponibilité de répondre à toutes mes questions. Merci de m'avoir soutenue tout au long de la préparation de ce mémoire.

Mes vifs remerciements vont aussi aux membres du jury qui m'ont fait l'honneur d'évaluer ce modeste travail.

Je témoigne ma reconnaissance aussi à tous les enseignants du département mathématiques de l'Université Mouloud MAMMERI qui m'ont guidée tout au long de mon cursus de formation.

Je remercie bien évidemment mes parents d'avoir été toujours là à mes côtés, de m'avoir encouragée tout au long de mes études. Sans eux je n'en serais sûrement pas arrivé là.

Je remercie, mes sœurs et frères qui m'ont toujours soutenue. Leurs présence a été pour moi salutaire dans les périodes de doute.

Je tiens à exprimer mes remerciements les plus profonds à tous les membres de ma famille, pour leurs soutiens indéfectibles.

En fin, mes remerciements s'adressent aussi à tous ceux et celles qui m'ont portée aide, encouragée et soutenue de près ou de loin, ainsi que mes amis(es) et camarades de promotion...

Table des matières

Table des figures

Introduction générale

Chaque problème qu'un être humain rencontre ne peut être suffisamment simple pour être compris sans avoir recours à un concept, idée ou notion. L'idée consiste à remplacer le phénomène qu'on étudie par un modèle de structure semblable, mais un peu moins complexe. Par conséquent les modèles sont une nécessité fondamentale de la démarche scientifique. Cependant dans la pratique trouver un modèle mathématique qui convienne à un problème donné n'est pas une tâche des plus faciles. L'idéal est de pouvoir prendre en charge les contraintes de problème réel et pouvoir les exprimer sous forme mathématique d'une manière adéquate et simple.

La programmation mathématique, et plus particulièrement l'optimisation vise à résoudre des problèmes où l'on cherche à déterminer parmi un grand nombre de solutions candidates celle qui donne le meilleur rendement. Plus précisément, on cherche à trouver une solution satisfaisant un ensemble de contraintes qui minimise ou maximise la fonction objectif donnée. L'application de la programmation mathématique est en expansion croissante et se retrouve dans plusieurs domaines.

Classification d'un programme mathématique

Voici les sections principales de la programmation mathématique devenues classiques.

Programmation linéaire

La fonction économique est linéaire, l'ensemble sur lequel on cherche l'extremum de cette fonction est donné par un système linéaire d'égalités et d'inégalités.

La programmation linéaire comporte des méthodes spéciales pour leur résolution, bien plus avantageuses que celles relatives aux problèmes de forme générale.

Programmation non linéaire

La fonction économique et/ou les contraintes sont non linéaires. La programmation non linéaire compte les divisions suivantes :

▷ **Programmation convexe**

La fonction économique et l'ensemble sur lequel on cherche a résoudre le problème d'extremum sont convexes.

▷ **programmation quadratique**

La fonction économique est quadratique, les contraintes étant des égalités et des inégalités linéaires.

▷ **Problèmes a extremums multiples**

On relève d'ordinaire dans ce domaine des classes spéciales des problèmes très fréquents dans les applications, par exemple, les problèmes relatifs à la minimisation des fonctions concaves sur un ensemble convexe.

Programmation en nombres entiers

C'est un problème d'optimisation dont toutes les variables sont contraintes à ne prendre que des valeurs entières.

Structurellement notre mémoire s'organise en trois chapitres :

Dans le premier chapitre, on représente certains fondements théoriques et généralités de la programmation linéaire.

Dans le deuxième chapitre, on représente une synthèse des méthodes de résolutions utilisées dans le domaine de la programmation linéaire mixte en nombres entiers, on étudie les deux principales familles de méthodes de résolutions : les méthodes de recherche arborescente et les méthodes des coupes.

Et finalement le troisième chapitre sera consacré à la représentation de logiciel Lingo, utile et résoudre quelques exemples retraités dans les chapitres précédents.

Enfin on clôture ce mémoire par une conclusion générale.

Chapitre 1

Rappels sur la programmation linéaire

1.1 Introduction

Le problème le plus simple de la programmation mathématique est celui de programmation linéaire.

La programmation linéaire peut être définie comme un outil mathématique qui permet d'analyser divers types de situations dans lesquelles nous retrouvons une fonction linéaire, appelée fonction objectif que l'on désire optimiser (maximiser ou minimiser), sous des contraintes linéaires.

1.2 Forme générale [2]

La forme générale d'un problème de programmation linéaire est :

$$\begin{cases} Opt & Z = C^t x \\ sc & Ax = b \ (\leq, \geq) \\ & x \geq 0 \end{cases}$$

$C \in \mathbb{R}^n$; $x \in \mathbb{R}^n$; A matrice (m, n) de rang $r(A)=m \leq n$; $b \in \mathbb{R}^m$

Il est possible de ramener le problème à des formes plus compactes mais équivalentes, en particulier aux formes dites "canoniques" et "standards" [9]

Ces deux formes équivalentes sont utilisées abondamment dans la suite :

La forme standard sera celle généralement utilisée pour la description des algorithmes.

La forme canonique sera particulièrement utile pour l'étude de la dualité.

La forme standard

$$\begin{cases} max \ (min) & Z = \sum_{j=1}^{n} C_j x_j \\ sc & \sum_{j=1}^{n} a_{ij} x_j = b_i \quad i = 1, ..., m \quad b_i \geq 0 \\ & x_j \geq 0 \quad j = 1, ..., n \end{cases}$$

La forme canonique

$$\begin{cases} max & Z = \sum_{j=1}^{n} C_j x_j \\ sc & \sum_{j=1}^{n} x_j \leq b_i \quad i = 1, ..., m \\ & x_j \geq 0 \quad j = 1, ..., n \end{cases}$$

$$\begin{cases} min & Z = \sum_{j=1}^{n} C_j x_j \\ sc & \sum_{j=1}^{n} x_j \geq b_i \quad i = 1, ..., m \\ & x_j \geq 0 \quad j = 1, ..., n \end{cases}$$

1.3 Classification des problèmes

Parmi les problèmes de programmation linéaire on trouve :

Les problèmes linéaires en variables mixtes :

Un problème linéaire en variables mixtes est un problème d'optimisation pour lequel certaines variables sont entières et d'autre sont réelles.

Les problèmes linéaires en nombres entiers :

Un problème linéaire en nombres entiers est un problème d'optimisation dont toutes les variables sont entières.

Les problèmes linéaires en variables binaires :

Un problème linéaire en variables binaires est un problème d'optimisation en nombres entiers, telle que les valeurs des variables entières sont dans l'ensemble $\{0, 1\}$.

1.4 Théorie sur la modélisation des problèmes

Un modèle est une construction mathématique utilisée pour représenter certains aspects significatifs de problèmes du monde réel.

Les étapes pour modéliser un problème sont les suivantes :

- ▷ Étape 1 : Déterminer les variables de décision et les formuler de façon algébrique : $x_1, ..., x_n$.
- ▷ Étape 2 : Déterminer les contraintes et les formuler comme des équations ou des inéquations dépendantes des variables de décision :

$$\begin{gathered} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \geq, \leq, = b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \geq, \leq, = b_2 \\ . \\ . \\ . \\ a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \geq, \leq, = b_m \end{gathered}$$

▷ Étape 3 : Présenter toutes les conditions implicitement établies conformément à la nature des variables : qu'elles ne peuvent pas être négatives, qu'elles soient entières, qu'elles ne peuvent que prendre des valeurs déterminées,...

$$x_1, ..., x_n \geq 0$$

$$x_1, ..., x_n \textit{ sont des nombres entiers, ou sont booleénnes, ...}$$

▷ Étape 4 : Déterminer la fonction objectif.

$$\textit{Maximiser ou minimiser } Z = C_1x_1 + ... + C_nx_n$$

Exemple :

On doit organiser un pont aérien pour transporter 1600 personnes et 90 tonnes de bagages. Les avions disponibles sont de deux types : 12 du type A et 9 du type B, le type A peut transporter à pleine charge 200 personnes et 6 tonnes de bagages, le type B peut transporter 100 personnes et 6 tonnes de bagages. La location d'un avion du type A coûte 80000 Euros, la location d'un avion du type B coûte 20000 Euros.

Le modèle mathématique de ce problème :

$$\begin{cases} min & Z = 80000x_1 + 20000x_2 \\ sc & 200x_1 + 100x_2 \leq 1600 \\ & 6x_1 + 6x_2 \leq 90 \\ & x_1 \leq 12 \\ & x_2 \leq 9 \\ & x_1, x_2 \geq 0 \;\; x_1, x_2 \textit{ entiers.} \end{cases}$$

avec x_i : nombres d'avions à louer.

1.5 Rappels d'algèbre linéaire [9]

1.5.1 Espace vectoriel

Un espace vectoriel E sur $\mathbb{R}$ est un ensemble d'éléments (appelés vecteurs ou points, et notés ici x, y, z ... vérifiant : $\lambda x + \mu y \in E \quad \forall\, x, y \in E \quad \forall\, \lambda, \mu \in \mathbb{R}$)

▷ 0 désigne le vecteur nul, neutre de la loi $+$.

▷ L'expression $\sum_j^r \lambda_j x^{(j)}$ est une combinaison linéaire des vecteurs $x^{(1)}$,..., $x^{(r)}$.

Si de plus $\sum_{j=1}^r \lambda_j = 1 \quad \lambda_j \geq 0$ la combinaison linéaire est appelée combinaison convexe.

▷ Les vecteurs $x^{(1)}, ..., x^{(r)}$ sont linéairement indépendants si :

$$\sum_{j=1}^r \lambda_j x^{(j)} = 0 \rightarrow \lambda_j = 0 \quad \forall j$$

▷ La dimension de E est le nombre maximale de vecteurs linéairement indépendants dans E.

Si E est de dimension n, une base de E est un ensemble de n vecteurs linéairement indépendants de E.

▷ Soit $x^{(1)}, ..., x^{(n)}$ une base de E. La représentation d'un vecteur quelconque x de E comme combinaison linéaire de vecteurs de la base est unique :

$$\exists\ \lambda_j \text{ unique } \mid\ x = \sum_{j=1}^n \lambda_j x^{(j)}$$

La substitution du vecteur x à un vecteur $x^{(l)}$ de la base fournit une nouvelle base si et seulement si $\lambda_l \neq 0$

1.5.2 Matrices

Une matrice $(A = a_{ij}; \quad i = 1, ..., m \quad j = 1, ..., n)$, de dimension $(m \times n)$, peut être considérée comme formée de m vecteurs linges de dimension n ;
$\alpha_i = (\alpha_{ij}; \quad j = 1, ..., n) \quad i = 1, ..., m$ ou de n vecteurs colonnes de dimension m;
$a_j = (a_{ij}; \quad i = 1, ..., m) \quad j = 1, ..., n$
Le nombre maximale de vecteurs linéairement indépendants parmi les m vecteurs lignes est égales au nombre maximale de vecteurs linéairement indépendants parmi les n vecteurs colonnes ; c'est ce nombre que l'on appelle le rang de la matrice A, noté $r(A)$.

Une matrice carrée B de dimension $(m \times m)$ est dite régulière si $r(A) = m$; dans ce cas, elle est inversible et la matrice inverse B^{-1}, de dimension $(m \times m)$ vérifie $B^{-1}B = BB^{-1} = I$ où I représente la matrice identité de dimension $(m \times m)$.

La matrice transposée de la matrice A de dimension $(m \times n)$ est notée A^t et est de dimension $(n \times m)$.

$$(A^t = a_{ji}; \quad j = 1, ..., n \quad i = 1, ..., m)$$

1.5.3 Systèmes d'équations

Soit un système de m équations à n inconnues :
$\sum_{j=1}^{n} a_{ij} x_j = b_i \quad i = 1, ..., m$
ou, en notation vectorielle, $Ax = b$

- ▷ Si :
  - . $r(A) < r(A, b)$: le système est impossible et ne possède aucune solution.
  - . $r(A) = r(A, b) = n$: le système possède une solution unique.
  - . $r(A) = r(A, b) < n$: le système possède une infinité de solutions.
- ▷ $r(A) = m$ (ce qui implique $m \leq n$), le système $Ax = b$ possède :
  - . une solution unique si $m = n$; cette solution est $x = A^{-1}b$
  - . une infinité de solution si $m < n$
- ▷ Si B une matrice régulière de dimension m, les systèmes $Ax = b$ et $BAx = Bb$ sont équivalents, c'est à dire qu'ils possèdent le même ensemble de solutions :

$$\{x \mid Ax = b\} = \{x \mid BAx = Bb\}$$

1.6 Polyèdres convexes

Un ensemble $A \subset \mathbb{R}^n$ est convexe si :

$$\alpha x + (1 - \alpha) y \in A \quad \forall\, x, y \in A \quad \forall\, \alpha \mid 0 \leq \alpha \leq 1$$

Un ensemble A convexe est caractérisé par le fait que toute combinaison convexe de points de A appartient à A.

$$\left.\begin{array}{c} x^{(1)}, ..., x^{(k)} \in A \\ \sum_{j=1}^{k} \alpha_j = 1 \quad \alpha_j \geq 0 \end{array}\right\} \rightarrow \sum_{j=1}^{k} \alpha_j x^{(j)} \in A$$

L'intersection de plusieurs ensembles convexes est un ensemble convexe.
Soit un système linéaire de m ($< \infty$) équations ou inéquations à n inconnues :

$$\sum_{j=1}^{n} a_{ij}x_j \leq b_i \quad i = 1, ..., m$$

Chacun des m ensembles $\{x \subset \mathbb{R}^n \mid \alpha_i x = b_i\} \quad i = 1, ..., m$ représente un hyperplan (une variété linéaire affine) de $\mathbb{R}^n$; chacun des m ensembles $\{x \subset \mathbb{R}^n \mid \alpha_i x \leq b_i\}$ $i = 1, ..., m$ représente un demi-espace fermé de $\mathbb{R}^n$ dont l'hyperplan correspondant constitue la frontière.

Le vecteur α_i est orthogonale à l'hyperplan $\alpha_i x = b_i$, étant donné :

$$\alpha_i(x^{(1)} - x^{(2)}) = 0 \quad \forall\, x^{(1)}, x^{(2)} \text{ dans l'hyperplan}$$

et est dirigé vers l'extérieur de demi-espace $\alpha_i x \leq b_i$ puisque $\alpha_i(y - x) \leq 0 \quad \forall y \in$ au demi-espace, $\forall x$ dans l'hyperplan.

Un polyèdre convexe P est l'intersection (éventuellement vide) d'un nombre fini de demi-espaces fermés ou d'hyperplans :

$$P = \{x \in \mathbb{R}^n \mid \alpha_i x \leq b_i \quad i = 1, ..., p \quad \alpha_i x = b_i \quad i = p+1, ..., m\}$$

Un polyèdre convexe P est borné s'il existe une valeur u finie et positive telle que :

$$|x_j| \leq u \quad \forall j\ ,\ \forall x \in P$$

1.7 Définition et terminologie

Soit :

$$(P)\begin{cases} opt\ Z = C^t x \\ sc\ \ Ax = b \\ x \geq 0 \end{cases}$$

$A(m, n)$; $r(A) = m \leq n$
$C \in \mathbb{R}^n$; $x \in \mathbb{R}^n$; $b \in \mathbb{R}^m$

Solution réalisable

Une solution réalisable est un vecteur $x \in \mathbb{R}^n$ qui vérifiant les contraintes de (P)
Exemple :

$$(P)\begin{cases} min & x_1 + x_2 \\ sc & 3x_1 + x_2 \leq 2 \\ & x_1 + 4x_2 \geq 3 \\ & x_1, x_2 \geq 0 \end{cases}$$

$x = (x_1, x_2) = (0, 1)$ est une solution réalisable de (P)
$x = (x_1, x_2) = (0, 0)$ n'est pas réalisable.

Solution optimale

Une solution est dite optimale si elle est réalisable et qu'elle optimise la fonction objectif.

Domaine réalisable

Le domaine réalisable est l'ensemble des solutions réalisables.

Base

C'est une sous matrice A_B d'ordre m extraite de A, de déterminant non nul $[detA_B \neq 0]$
Les vecteurs colonnes de A_B sont linéairement indépendant.
Les variables associées aux colonnes de A_B sont appelées variables de base.
Les $n-m$ variables restantes sont appelées variables hors-base.
Dans ce cas A et x sont décomposés de la manière suivante :

$$A = [A_B, A_H] \quad x = \begin{pmatrix} x_B \\ x_H \end{pmatrix}$$

$$\begin{aligned} Ax = b &\Longleftrightarrow [A_B, A_H]\begin{pmatrix} x_B \\ x_H \end{pmatrix} = b \\ &\Longleftrightarrow A_B x_B + A_H x_H = b \end{aligned}$$

Solution de base

C'est la solution obtenue en annulant les variables hors base (en posant $x_H = 0$)
$A_B x_B = b \iff x_B = A^{-1}b$.

Exemple

$$\begin{cases} max\ Z & x_1 + 3x_2 \\ sc & x_1 + x_2 \leq 14 \\ & -2x_1 + 3x_2 \leq 12 \\ & 2x_1 - x_2 \leq 12 \\ & x_1, x_2 \geq 0 \end{cases}$$

La forme standard

$$\begin{cases} max\ Z & x_1 + 3x_2 \\ sc & x_1 + x_2 + x_3 = 14 \\ & -2x_1 + 3x_2 + x_4 = 12 \\ & 2x_1 - x_2 + x_5 = 12 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

$$A = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ -2 & 3 & 0 & 1 & 0 \\ 2 & -1 & 0 & 0 & 1 \end{pmatrix}$$

$rg\,(A) = 3$

Si $A_B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$, alors $x = (x_1, x_2, x_3, x_4, x_5) = (0, 0, 14, 12, 12)$ est la solution de base.

Si $A_B = \begin{pmatrix} 1 & 1 & 1 \\ -2 & 3 & 0 \\ 2 & -1 & 0 \end{pmatrix}$

$det\,(A) \neq 0$

$A_B^{-1} = \begin{pmatrix} 0 & \frac{1}{4} & \frac{3}{4} \\ 0 & \frac{1}{2} & \frac{1}{2} \\ 1 & -\frac{3}{4} & -\frac{5}{4} \end{pmatrix}$

donc la solution de base est $x_B = A^{-1}b \iff x_B = (12, 12, -10, 0, 0)$

Solution de base réalisable

Une solution de base est réalisable si et seulement si toutes les composantes de base sont positives ou nulles c'est à dire $x_B = A_B^{-1}b \geq 0$.

Solution de base réalisable non dégénérée

Une solution de base réalisable est dite non dégénérée si toutes les composantes de x_B sont strictement positives $x_B = A_B^{-1}b > 0$.

La dégénérescence est un phénomène fréquent dans certains types de problèmes (problème de transport, problème de flots, de plus court chemin...) [9].

1.8 Caractérisation algébrique des points extrêmes

Définition :

Un polyèdre convexe, borné et non vide, est appelé un **polytope.**

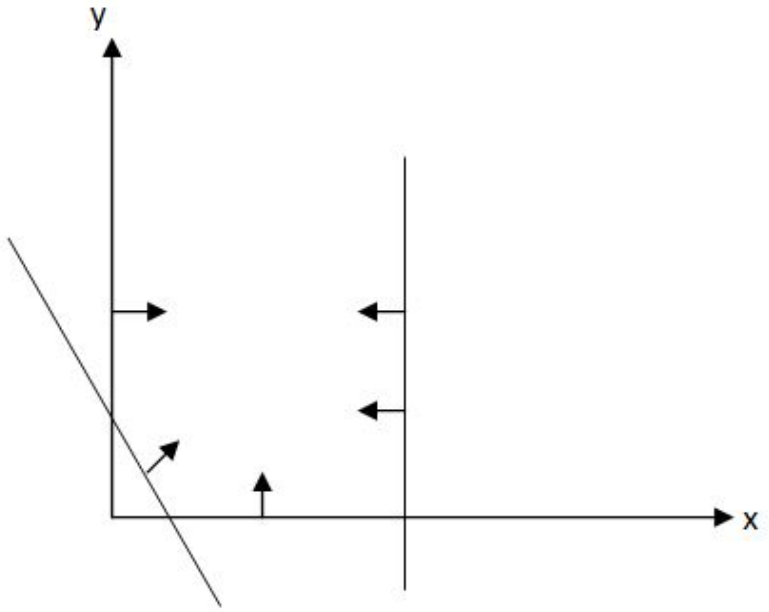


FIGURE 1.1 – Polyèdre non borné

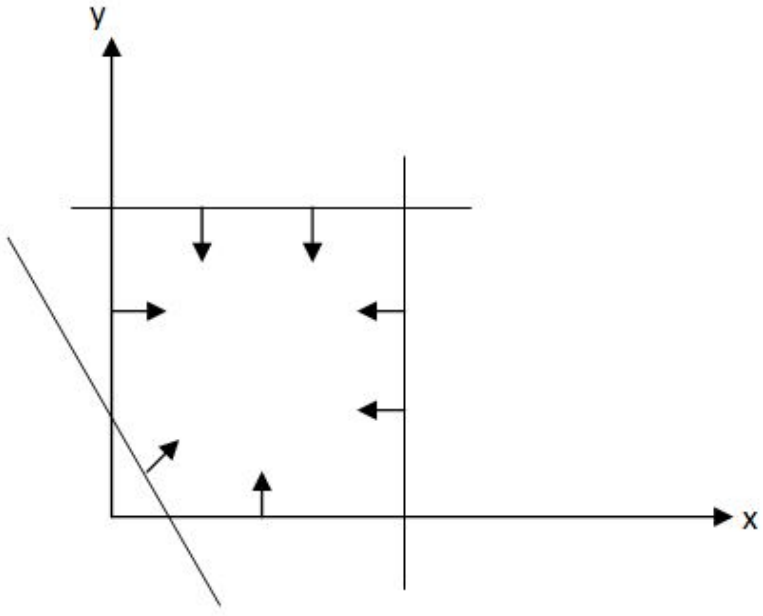


FIGURE 1.2 – Polytope

Définition :

On appelle sommet de X (ou point extrême) tout points de X qui ne peut s'écrire comme combinaison convexe d'autre points de X.

Théorème [14] :

- ▷ L'ensemble des points extrêmes d'un polyèdre $X = \{x \in \mathbb{R}^n \mid Ax = b \quad x \geq 0\}$ correspond à l'ensemble des solutions de base réalisable.
- ▷ Tout point x de X peut s'écrire comme combinaison convexe des sommets de X.

Théorème [14] :

L'optimum de la fonction économique Z de (P) est atteint en un sommet du domaine réalisable X de (P), si l'optimum est atteint en plusieurs sommets, alors il est atteint en tout points combinaisons convexe de ces sommets.

1.9 Méthodes de résolution

La résolution de différentes sortes de problèmes rencontrés dans notre vie quotidienne a poussé les chercheurs à proposer des méthodes de résolution et à réaliser de grands efforts pour améliorer leurs performances en termes de temps de calcul nécessaire et de la qualité de la solution proposée. Au fil des années, de nombreuses méthodes de résolution de problèmes de différentes complexités ont été proposées. Les méthodes de résolution exactes sont nombreuses et se caractérisent par le fait qu'elles permettent d'obtenir une ou plusieurs solutions dont l'optimalité est garantie. Parmi ces méthodes, on peut citer la méthode graphique, l'algorithme du simplexe et l'algorithme dual du simplexe.

1.9.1 Méthode graphique

La méthode graphique permet la résolution de problèmes linéaires simples.

Cette méthode est limitée aux problèmes à deux ou trois variables de décision puisqu'il n'est pas possible d'illustrer graphiquement plus de trois dimensions. Bien qu'en général on puisse difficilement trouver des problèmes avec seulement deux ou trois variables de décision, cette méthodologie de résolution est cependant très utile. Lors de la reproduction graphique de situations possibles, tels que :

- ▷ L'existence d'une solution optimale unique, des solutions optimales alternatives.
- ▷ La non existence de solution et l'absence de partie bornée.

La méthode graphique nous offre une aide visuelle pour interpréter et comprendre l'algorithme de la méthode du Simplexe.

Les étapes de résolution des problèmes linéaires par la méthode graphique sont les suivantes :

- ▷ Créer un système de coordonnées cartésiennes, dans lequel chaque variable de décision est représentée par un axe.
- ▷ Établir une échelle de mesure pour chacun des axes appropriés à sa variable associée.

- ▷ Dessiner dans le système de coordonnées les contraintes du problème, y compris celles des variables de décision. .
- ▷ Remarquer qu'une inéquation précise une région qui sera le demi-plan limité par la ligne droite, qui représente la contrainte qu'on considère comme une contrainte d'égalité alors que si une équation linéaire détermine une région c'est la ligne droite, elle-même.
- ▷ L'intersection de toutes les régions détermine la région ou l'espace faisable (qui est un ensemble convexe). Si non, si il n'y a pas de point qui satisfait toutes les contraintes simultanément de sorte que le problème ne sera pas résolu, on dit que le problème est infaisable.
- ▷ Déterminer les points extrêmes ou les sommets du polyèdre qui forme la région faisable.
- ▷ Ces points seront les candidats à la solution optimale.
- ▷ Évaluer la fonction objectif à chaque sommet et celui qui maximise la fonction objectif définie la solution optimale (pour un problème de maximisation).

Exemple

soit le PL suivant :

$$(P)\begin{cases} max \quad Z = 1200x_1 + 1000x_2 \\ \qquad 3x_1 + 4x_2 \leq 160 \\ \qquad 6x_1 + 3x_2 \leq 180 \\ \qquad x_1, x_2 \geq 0 \end{cases}$$

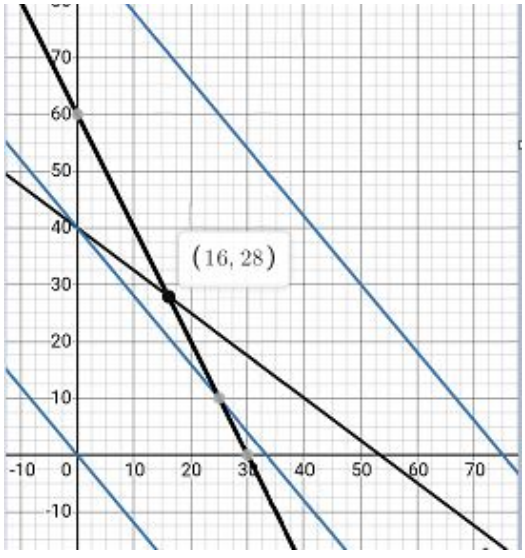


FIGURE 1.3 – La représentation graphique

La solution optimale est $x = (x_1, x_2) = (16, 28)$, d'où $Z = 47200$.

1.9.2 Méthode du simplexe [9],[14]

L'algorithme dit **du simplexe** pour la résolution des programmes linéaires est largement du a Dantzig (dont les premiers travaux se situent vers la fin des années 1940) bien que l'on puisse remonter a Fourier (1826) pour un premier énoncé des principes de base [14].

La méthode du simplexe est une méthode efficace pour résoudre les problémes linéaires, elle peut s'appliquer peu importe le nombre de variables dans le modèle, elle consiste a déplacer d'un sommet a un autre tout en améliorant la valeur de la fonction économique.

Son application nécessite la connaissance :

- ▷ Des formules de changement de base
- ▷ De la formule de variation de l'objectif
- ▷ Des critères de choix pour le changement de base
- ▷ D'un critère d'optimalité
- ▷ Des critères pour le cas ou la solution infinie

Principe de l'algorithme du simplexe

La recherche systématique d'une solution optimale à l'aide de l'algorithme de simplexe peut se résumer comme suit :

1. Déterminer une première solution de base réalisable, cette solution **initiale** sert comme point de départ vers la solution optimale si elle existe.
2. Si la solution obtenue en (1) n'est pas optimale, déterminer une autre solution de base réalisable qui permettrait d'améliorer la fonction objectif (augmentation pour une maximisation ou diminution pour une minimisation).
3. On répète cette procédure itérative jusqu'à ce qu'il ne soit plus possible d'améliorer la fonction objectif. La dernière solution de base réalisable obtenue constitue la solution optimale du programme linéaire.

L'algorithme nous permettra, avec divers critères, de détecter si la solution optimale est unique ou s'il existe des solutions optimales multiples ou encore si le modèle ne permet pas d'obtenir une valeur finie pour la fonction objectif.

Formules de changement de base

Soit le programme suivant :

$$(P)\begin{cases} max\ Z & = Cx \\ sc & Ax = b \\ & x \geq 0 \end{cases}$$

Supposons qu'il existe une base explicite B, on note par $I(B) = \{1, 2, 3, ..., m\}$ l'ensemble des indices de base, et $I(H)$=$\{m+1, m+2, ..., n\}$ l'ensemble des indices hors base.

Les vecteurs de base sont :

$$V_1 = \begin{pmatrix} 1 \\ 0 \\ . \\ . \\ 0 \end{pmatrix} ; \qquad V_2 = \begin{pmatrix} 0 \\ 1 \\ . \\ . \\ 0 \end{pmatrix} ; \qquad ... \qquad ; V_m = \begin{pmatrix} 0 \\ 0 \\ . \\ . \\ 1 \end{pmatrix}$$

$$\forall\, j \in I(H)\ ; \quad V_j = \begin{pmatrix} a_{1j} \\ a_{2j} \\ . \\ . \\ a_{mj} \end{pmatrix}$$

$$V_j = a_{1j} \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \\ . \\ 0 \end{pmatrix} + a_{2j} \begin{pmatrix} 0 \\ 1 \\ 0 \\ . \\ . \\ 0 \end{pmatrix} + ... + a_{mj} \begin{pmatrix} 0 \\ 0 \\ 0 \\ . \\ . \\ 1 \end{pmatrix}$$

$V_j = \sum_{i \in I(B)} a_{ij} V_i\ ;\ \forall j \in I(H)$

$V_0 = \begin{pmatrix} b_1 \\ b_2 \\ . \\ . \\ b_m \end{pmatrix}$; le vecteur second membre du système $Ax = b$

$$V_0 = b_1 \begin{pmatrix} 1 \\ 0 \\ 0 \\ . \\ . \\ 0 \end{pmatrix} + b_2 \begin{pmatrix} 0 \\ 1 \\ 0 \\ . \\ . \\ 0 \end{pmatrix} + ... + b_m \begin{pmatrix} 0 \\ 0 \\ . \\ . \\ 0 \\ 1 \end{pmatrix}$$

$V_0 = \sum_{i \in I(B)} b_i V_i$

$$V_k = \begin{pmatrix} a_{1k} \\ a_{2k} \\ . \\ . \\ a_{rk} \\ . \\ . \\ a_{mk} \end{pmatrix}$$

$V_k = \sum_{i \in I(B)} a_{ik} V_i + a_{rk} V_r$

$$\Longrightarrow V_r = \frac{V_k}{a_{rk}} - \sum_{i \in I(B)} \frac{a_{ik} V_i}{a_{rk}} \text{ pour } i \neq r$$

$$\forall\ j\in I(H)\,;\, V_j = \sum_{i\in I(B)} a_{ij}V_i = \sum a_{ij}V_i + a_{rj}V_r$$

$$= \sum_{i\in I(B)} a_{ij}V_i + a_{rj}\left(\frac{V_k}{a_{rk}} - \sum_{i\in I(B)} \frac{a_{ik}}{a_{rk}}V_i\right);\ pour\ i \neq r$$

$$= \sum_{i\in I(B)}\left(a_{ij} - a_{rj}\frac{a_{ik}}{a_{rk}}\right)V_i + \frac{a_{rj}}{a_{rk}}V_k$$

$$= \sum_{i\in(B')} a'_{ij}V_i + a'_{kj}V_k$$

$$= \sum_{i\in I(B')} a'_{ij}V_i$$

$$\boxed{\begin{array}{c}\text{Où}\quad a'_{ij} = a_{ij} - a_{rj}\frac{a_{ik}}{a_{rk}} \quad i \neq k \\ a'_{kj} = \frac{a_{rj}}{a_{rk}}\end{array}}$$

$$V_0 = \sum_{i\in I(B)} b_iV_i = \sum_{i\in I(B)} b_iV_i + b_rV_r \quad i \neq r$$

$$= \sum b_iVi + b_r\left(\frac{V_k}{a_{rk}} - \sum_{i\in I(B)} \frac{a_{ik}}{a_{rk}}V_i\right)$$

$$= \sum_{i\in I(B)}(b_i - \frac{b_r}{a_{rk}}a_{ik})V_i + \frac{b_r}{a_{rk}}V_k \quad i \neq k$$

$$= \sum_{i\in I(B')} b'_iV_i + b'_kV_k = \sum_{i\in I(B')} b'_iV_i \quad i \neq k$$

$$\boxed{\begin{array}{c}b'_i = b_i - a_{ik}\frac{b_r}{a_{rk}} \quad i \neq k \\ b'_k = \frac{b_r}{a_{rk}}\end{array}}$$

Variation de la fonction économique

On note par :

$$Z_0 = \sum_{i\in I(B)} C_ib_i \ \text{ la valeur initiale de } Z$$

$$I(B') = (I(B)\setminus\{r\}) \cup \{k\}$$

La nouvelle valeur de Z sera :

$$Z'_0 = \sum_{i\in I(B')} C_ib'_i$$

$$= \sum_{i\in I(B)} C_ib'_i - C_rb'_r + C_kb'_k$$

$$= \sum_{i\in I(B)} C_ib'_i + C_kb'_k$$

$$= \sum_{i\in I(B)} C_i\left(b_i - \frac{b_r}{a_{rk}}a_{ik}\right) + C_k\frac{b_r}{a_{rk}}$$

$$= \sum_{i\in I(B)} C_ib_i - \frac{b_r}{a_{rk}}\left(\sum_{i\in I(B)} C_ia_{ik} - C_k\right)$$

$$= Z_0 - \frac{b_r}{a_{rk}}(Z_k - C_k)$$

$$\boxed{Z'_0 = Z_0 - \frac{b_r}{a_{rk}}(Z_k - C_k)}$$

On note par :

$$\Delta_j = Z_j - C_j = \sum_{i \in I(B)} C_i a_{ij} - C_j$$

la nouvelle valeur de Δ_j est :

$$\begin{aligned}
\Delta'_j &= \sum_{i \in I(B')} C_i a'_{ij} - C_j \\
&= \sum_{i \in I(B)} C_i a'_{ij} - C_r a'_{rj} + C_k a'_{kj} - C_j \\
&= \sum_{i \in I(B)} C_i a'_{ij} + C_k a'_{kj} - C_j \\
&= \sum_{i \in I(B)} C_i (a_{ij} - \frac{a_{rj}}{a_{rk}} a_{ik}) + C_k \frac{a_{rj}}{a_{rk}} - C_j \\
&= (\sum C_i a_{ij} - C_j) - \frac{a_{rj}}{a_{rk}} (\sum C_i a_{ik} - C_k)
\end{aligned}$$

$$\boxed{\Delta'_j = \Delta_j - \frac{a_{rj}}{a_{rk}} \Delta_k}$$

Critère d'optimalité

Théorème :

Si pour une base donnée tous les $\Delta_j \geq 0$, alors la solution de base correspondante réalise le maximum de la fonction Z.

Choix de la variable entrante x_k

On choisit l'indice k tq :

$$\Delta_k = min_{\Delta_j < 0} \Delta_j$$

Choix de variable sortante

On choisit l'indice k tq :

$$\frac{b_r}{a_{rk}} = min_{a_{ik} > 0} \frac{b_i}{a_{ik}}$$

Remarque :

S'il existe un indice $k \in I(H)$ tq : $\Delta_k < 0$ et $a_{ik} \leq 0 \; \forall i$; alors la solution du problème est infinie ($Z \to +\infty$)

Tableau du simplexe

			C_1	C_2	...	C_r	C_m	C_{m+1}	.....	C_k	....	C_n
C_b	base	b	x_1	x_2	...	x_r	x_m	x_{m+1}	.....	x_k	....	x_n
C_1	x_1	b_1	1	0	...	0	0	$a_{1,m+1}$	.....	a_{1k}	.....	a_{1n}
C_2	x_2	b_2	0	1		0	...	$a_{2,m+1}$	.....	a_{2k}		a_{2n}
...	...	...	...	0		0	...	.....	.....	.....	.....	.....
...	...	...	...	...		...	...	.....	.....	.....	.....	.....
C_r	x_r	b_r	0	0	...	1	...	$a_{r,m+1}$	.....	a_{rk}	.....	a_{rn}
...	...	...	...	...		0	...	.....	.....	.....	.....	.....
...	...	...	...	...		...	...	.....	.....	.....	.....	.....
C_m	x_m	b_m	0	0	...	...	1	$a_{m,m+1}$	.....	a_{mk}	.....	a_{mn}
			0	0	...	0	0	Δ_{m+1}	....	Δ_k	....	Δ_n

Remarque :

$min\ Z = -max\ (-Z)$

Organigramme de la méthode du simplexe :

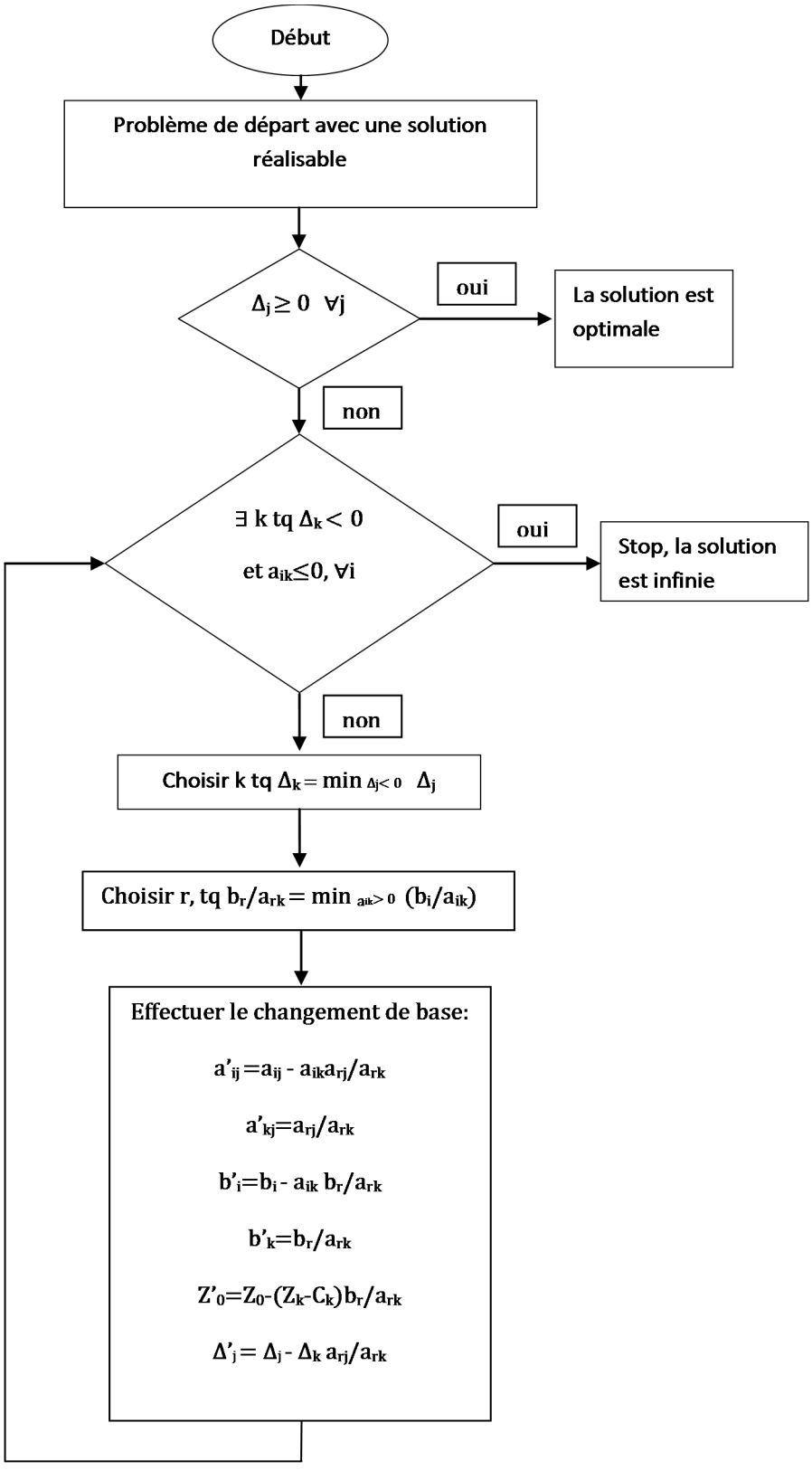


FIGURE 1.4 – Organigramme de la méthode du simplexe

Exemple :

$$\begin{cases} max\, Z & = 3x_1 + 5x_2 \\ sc & 3x_1 + 2x_2 \leq 18 \\ & x_1 \leq 4 \\ & x_2 \leq 6 \\ & x_1, x_2 \geq 0 \end{cases}$$

la forme standard

$$\begin{cases} max\, Z & = 3x_1 + 5x_2 + 0t_1 + 0t_2 + 0t_3 \\ sc & 3x_1 + 2x_2 + t_1 = 18 \\ & x_1 + t_2 = 4 \\ & x_2 + t_3 = 6 \\ & x_1, x_2, t_1, t_2, t_3 \geq 0 \end{cases}$$

Tableau 1

C_B	base	b	x_1	x_2	t_1	t_2	t_3
0	t_1	18	3	2	1	0	0
0	t_2	4	1	0	0	1	0
0	$\leftarrow t_3$	6	0	$\underline{1}$	0	0	1
		$Z = 0$	-3	$\uparrow$-5	0	0	0

Comme les Δ_j des variables hors base ne sont pas tous ≥ 0, cette base n'est pas optimale. on va donc effectuer un déplacement par changement de base.

Choisissons comme variable entrant dans la nouvelle base, la variable hors base ayant Δ_j le plus négatif, c'est à dire x_2, la variable t_3 va quitter la base.

Tableau 2

C_B	base	b	x_1	x_2	t_1	t_2	t_3
0	$\leftarrow t_1$	6	$\underline{3}$	0	1	0	-2
0	t_2	4	1	0	0	1	0
5	x_2	6	0	1	0	0	1
		$Z = 30$	$\uparrow$-3	0	0	0	5

La nouvelle solution de base correspondante est :

$$t_1 = 6,\ t_2 = 4,\ x_2 = 6$$

$$x_1 = 0,\ t_3 = 0;$$

$x = (0, 6)$ de coût $Z = 30$

Cette solution n'est toujours pas optimale, x_1 va donc rentrer en base, la variable t_1 va quitter la base.

Tableau 3

C_B	base	b	x_1	x_2	t_1	t_2	t_3
3	x_1	2	1	0	$\frac{1}{3}$	0	$\frac{-2}{3}$
0	t_2	2	0	0	$-\frac{1}{3}$	1	$\frac{2}{3}$
5	x_2	6	0	1	0	0	1
		$Z = 36$	0	0	1	0	3

La nouvelle solution de base correspondante est :

$$x_1 = 2,\ t_2 = 2,\ x_2 = 6$$

$$t_1 = 0,\ t_3 = 0;$$

$$x = (2, 6)$$

de coût 36, comme les Δ_j de toutes les variables sont ≥ 0, cette solution est donc optimale, et les itérations se terminent.

Algorithme de simplexe à l'aide des variables artificielles

En regardant la forme canonique d'un PL, nous remarquons dans les contraintes l'existence des inégalités d'infériorités ' $\leq$ ' et des variables de décisions non négatives.

Souvent, on commence l'algorithme de simplexe au sommet $x = (0, 0, \ldots, 0)$ qui appartient à l'espace de solution réalisable. Si la forme canonique n'est pas respectée et s'il existe une contrainte de supériorité ' $\geq$ ' on remarque ce qui suit :

En premier temps, pour atteindre la forme standard on ajoute des variables d'écart non négatives. Puisqu'on a une inégalité ' $\geq$ ', ces variables d'écart sont d'un signe négatif.

$Ex :\ x_1 + 3x_2 \ \geq\ 4 \quad (x_1, x_2 \geq 0)$,

Devient : $x_1 + 3x_2 – t_1 = 4 \quad (x_1, x_2, t_1 \geq 0)$

Si on souhaite initier l'algorithme de simplexe à l'origine $(0, 0)$ on aura :

$-t_1 = 4$ càd $t_1 = –4$ contradiction avec la contrainte de non négativité de t_1.

Donc le point origine $x = (0, 0, \ldots, 0)$ ne peut pas appartenir à l'espace de solution réalisable, d'où le lancement de l'algorithme avec une solution artificielle, d'où l'introduction aux équations semblables, des variables dites **artificielle**s pour contourner le problème.

Pour résoudre le nouveau PL, deux méthodes étroitement liées sont les plus citées dans la littérature : les M-Méthodes et le simplexe en 2-Phases.

M-Méthodes (appelées aussi Méthodes des grands M)

Organigramme de la M-méthode :

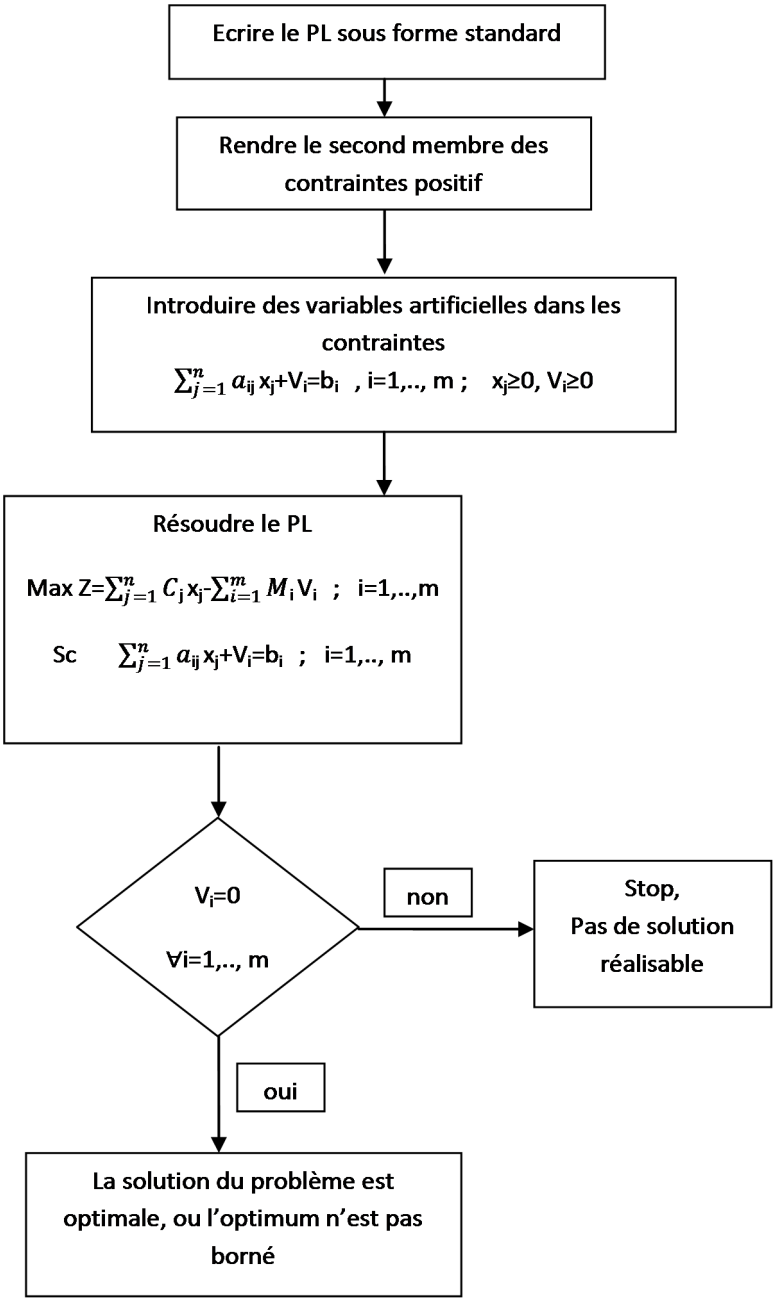


FIGURE 1.5 – Organigramme de la M-méthode

Exemple

Soit le PL suivant :

$$(P)\begin{cases} min & Z = x_1 + x_2 \\ sc & 2x_1 - x_2 \geq 2 \\ & -x_1 + x_2 \geq -2 \\ & x_1 + x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{cases}$$

La forme standard

$$(P)\begin{cases} min & Z = x_1 + x_2 + 0t_1 + 0t_2 + 0t_3 \\ sc & 2x_1 - x_2 - t_1 = 2 \\ & -x_1 + x_2 - t_2 = -2 \\ & x_1 + x_2 + t_3 = 5 \\ & x_1, x_2 \geq 0, t_1, t_2, t_3 \geq 0 \end{cases}$$

▷ rendre le seconde membre positif :

$$(P)\begin{cases} min & Z = x_1 + x_2 + 0t_1 + 0t_2 + 0t_3 \\ sc & 2x_1 - x_2 - t_1 = 2 \\ & x_1 - x_2 + t_2 = 2 \\ & x_1 + x_2 + t_3 = 5 \\ & x_1, x_2 \geq 0, t_1, t_2, t_3 \geq 0 \end{cases}$$

▷ Introduction d'une variable artificielle :

$$(P)\begin{cases} min & Z = x_1 + x_2 + 0t_1 + 0t_2 + 0t_3 + MV_1 \\ sc & 2x_1 - x_2 - t_1 + V_1 = 2 \\ & x_1 - x_2 + t_2 = 2 \\ & x_1 + x_2 + t_3 = 5 \\ & x_1, x_2 \geq 0, t_1, t_2, t_3 \geq 0 \end{cases}$$

Tableau 1

			1	1	0	0	0	M
C_B	base	b	x_1	x_2	t_1	t_2	t_3	V_1
M	$\leftarrow V_1$	2	$\underline{2}$	-1	-1	0	0	1
0	t_2	2	1	-1	0	1	0	0
0	t_3	5	1	1	0	0	1	0
			$\uparrow 2M-1$	$-M-1$	$-M$	0	0	0

Tableau 2

C_B	base	b	x_1	x_2	t_1	t_2	t_3
1	x_1	1	1	$\frac{-1}{2}$	$\frac{-1}{2}$	0	0
0	t_2	1	0	$\frac{-1}{2}$	$\frac{1}{2}$	1	0
0	t_3	4	0	$\frac{3}{2}$	$\frac{1}{2}$	0	1
			0	$\frac{-3}{2}$	$\frac{-1}{2}$	0	0

La solution optimale est $x = (1, 0, 0, 1, 4)$

La méthode des deux phases

Organigramme de la méthode de deux phases :

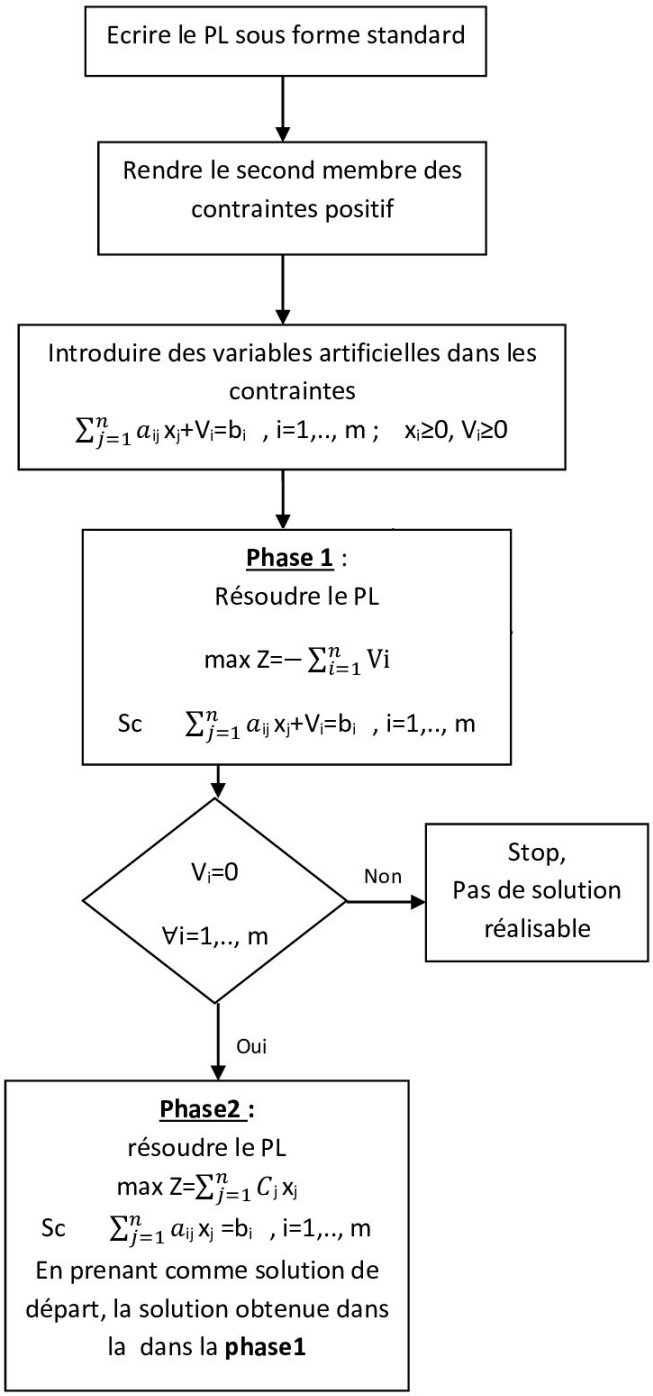


FIGURE 1.6 – Organigramme de la méthode de deux phases

Exemple

Soit à résoudre le PL suivant :

$$(P)\begin{cases} max & Z = -3x_1 + x_2 \\ sc & x_1 + x_2 \geq 1 \\ & 2x_1 + 3x_2 \geq 2 \\ & x_1, x_2 \geq 0 \end{cases}$$

La forme standard

$$(P)\begin{cases} max & Z = -3x_1 + x_2 + 0t_1 + 0t_2 \\ sc & x_1 + x_2 - t_1 = 1 \\ & 2x_1 + 3x_2 - t_2 = 2 \\ & x_1, x_2 \geq 0;\ t_1, t_2 \geq 0 \end{cases}$$

▷ Introduction des variables artificielles :

$$(P)\begin{cases} max & Z = -3x_1 + x_2 + 0t_1 + 0t_2 - V_1 - V_2 \\ sc & x_1 + x_2 - t_1 + V_1 = 1 \\ & 2x_1 + 3x_2 - t_2 + V_2 = 2 \\ & x_1, x_2 \geq 0;\ t_1, t_2 \geq 0;\ V_1, V_2 \geq 0 \end{cases}$$

Phase 1

Résoudre le PL suivant :

$$\begin{cases} max & Z = -V_1 - V_2 \\ sc & x_1 + x_2 - t_1 + V_1 = 1 \\ & 2x_1 + 3x_2 - t_2 + V_2 = 2 \\ & x_1, x_2 \geq 0;\ t_1, t_2 \geq 0;\ V_1, V_2 \geq 0 \end{cases}$$

Tableau 1

C_B	base	b	x_1	x_2	t_1	t_2	V_1	V_2
-1	V_1	1	1	1	-1	0	1	0
-1	$\leftarrow V_2$	2	2	3	0	-1	0	1
			-3	$\uparrow$-4	1	1	0	0

Tableau 2

C_B	base	b	x_1	x_2	t_1	t_2	V_1
-1	$\leftarrow V_1$	$\frac{1}{3}$	$\frac{1}{3}$	0	-1	$\frac{1}{3}$	1
0	x_2	$\frac{2}{3}$	$\frac{2}{3}$	1	0	-$\frac{1}{3}$	0
			-$\frac{1}{3}$	0	1	$\uparrow$-$\frac{1}{3}$	0

Tableau 3

C_B	base	b	x_1	x_2	t_1	t_2
0	t_2	1	1	0	-3	1
0	x_2	1	1	1	-1	0
			0	0	0	0

La solution de la base réalisable obtenue est : $x = (0, 1, 0, 1)$

Phase 2

Résoudre le PL suivant :

$$\begin{cases} max & Z = -3x_1 + x_2 + 0t_1 + 0t_2 \\ sc & x_1 + x_2 - t_1 = 1 \\ & 2x_1 + 3x_2 - t_2 = 2 \\ & x_1, x_2 \geq 0;\ t_1, t_2 \geq 0 \end{cases}$$

Tableau 1

C_B	base	b	x_1	x_2	t_1	t_2
0	t_2	1	1	0	-3	1
1	x_2	1	1	1	-1	0
			4	0	-1	0

Pas de pivot positif, l'optimum est non borné ($Z \to +\infty$)

1.10 La notion de dualité

- ▷ Pour éviter l'utilisation des variables artificielles, quand on a un problème avec des contraintes sous forme : $\sum a_{ij}x_j \geq b_i \quad i = 1, ..., m$; on utilise l'algorithme dual du simplexe.
- ▷ A chaque modèle de programmation linéaire correspond un autre modèle appelé **"dual".**
- ▷ La notion de dualité est un concept fondamental en programmation linéaire et conduit à un résultat important d'un point de vue théorique et pratique : **le théorème de la dualité.**

Dual d'un PL sous forme canonique

$$(P_1)\begin{cases} max & Z = C^t x \\ sc & Ax \leq b \\ & x \geq 0 \end{cases} \qquad (D_1)\begin{cases} min & \omega = Y^t b \\ sc & YA \geq C \\ & Y \geq 0 \end{cases}$$

$$(P_2)\begin{cases} min & Z = C^t x \\ sc & Ax \geq b \\ & x \geq 0 \end{cases} \qquad (D_2)\begin{cases} max & \omega = Y^t b \\ sc & YA \leq C \\ & Y \geq 0 \end{cases}$$

Définition du dual dans le cas général [14],[1]

On peut définir le dual d'un programme linéaire quelconque sans passer par la forme canonique, le tableau suivant résume les correspondances entre le primal et le dual :

Primal	Dual
Fonction économique max	second membre de contrainte i
Second membre b_i	Fonction économique min
A matrice des contraintes	A^t matrice des contraintes
nombre de contraintes	nombre de variables
contraintes i $\leq$	variable $y_i \geq 0$
contraintes i $\geq$	variable $y_i \leq 0$
contraintes i $=$	variable y_i $S.R.S$
nombre de variables	nombre de contraintes
variable $x_j \geq 0$	contrainte $j : \geq$
variable $x_j \leq 0$	contrainte j : $\leq$
variable x_j $S.R.S$	contrainte j : $=$

(S.R.S = sans restriction de signe)

Exemple

$$(P)\begin{cases} min & Z = 2x_1 - 3x_2 \\ & x_1 - x_2 \leq 1 \\ & 2x_1 + 3x_2 \geq 4 \\ & x_1 + x_2 = 3 \\ & x_1 \geq 0, x_2 \ S.R.S \end{cases}$$

$$(D)\begin{cases} max & \omega = y_1 + 4y_2 \\ & y_1 + 2y_2 + y_3 \leq 2 \\ & -y_1 + 3y_2 + y_3 = -3 \\ & y_1 \leq 0, y_2 \geq 0, y_3 \ S.R.S \end{cases}$$

Remarque :

Le dual du dual est le primal.

1.10.1 Théorèmes de la dualité [14]

Soit les problèmes suivant :

$$(P)\begin{cases} max & Z = C^t x \\ sc & Ax \leq b \\ & x \geq 0 \end{cases} \qquad (D)\begin{cases} min & \omega = Y^t b \\ sc & YA \geq C \\ & Y \geq 0 \end{cases}$$

Théorème 1 :

Si x^*et Y^* deux solutions réalisables pour (P) et (D) respectivement, alors :

$Z^* = Cx^* \leq Y^*b = \omega^*$

Théorème 2 :

Si x^*et Y^* deux solutions réalisables pour (P) et (D) respectivement vérifiant $Cx^* = Y^*\omega$, alors x^* est optimale pour (P) et Y^* est optimale pour (D)

Théorème 3 :

- ▷ Si la solution optimale du primal existe et est finie, alors la solution optimale du dual existe et est finie.
- ▷ Si l'un du primal ou du dual à un optimum non borné (solution infinie), alors l'autre n'a pas de solution (les contraintes sont contradictoires).

1.10.2 Théorème sur les écarts complémentaires [9],[14]

Deux solutions x^* *et* Y^* du primal et du dual respectivement sont optimales si et seulement si :

$$Y^{t*}(b - Ax^*) = 0$$

$$(A^tY - C)^t x^* = 0$$

Exemple :

Soit le problème suivant :

$$(P)\begin{cases} max \quad Z = 4x_1 + 6x_2 + 20x_3 + 17x_4 \\ x_1 + x_3 + 2x_4 \leq 10 \\ x_2 + 2x_3 + x_4 \leq 4 \\ x_1, x_2, x_3, x_4 \geq 0 \end{cases}$$

Vérifions si $x = (2, 0, 0, 4)$ et $Y = (4, 9)$ sont optimales pour (P) et son dual (D) :

$$Y(b - Ax) = (4,9)\left[\begin{pmatrix}10\\4\end{pmatrix} - \begin{pmatrix}1&0&1&2\\0&1&2&1\end{pmatrix}\begin{pmatrix}2\\0\\0\\4\end{pmatrix}\right]$$

$$=(4,9)\left[\begin{pmatrix}10\\4\end{pmatrix} - \begin{pmatrix}10\\4\end{pmatrix}\right] = 0$$

$$(YA - C)x = \left[(4,9)\begin{pmatrix}1&0&1&2\\0&1&2&1\end{pmatrix} - (4,6,20,17)\right]\begin{pmatrix}2\\0\\0\\4\end{pmatrix}$$

$$= [(4, 9, 22, 17) - (4, 6, 20, 17)] \begin{pmatrix} 2 \\ 0 \\ 0 \\ 4 \end{pmatrix}$$

$$= 0$$

Donc $x = (2, 0, 0, 4)$ et $y = (4, 9)$ sont optimales pour (P) et (D) respectivement.

1.10.3 Algorithme dual simplexe [9]

Soit le PL suivant :

$$(P) \begin{cases} min & Z = Cx \\ sc & Ax \geq b \\ & x \geq 0 \end{cases}$$

L'algorithme dual simplexe :

1. Écrire le PL sous forme standard
2. Rendre positifs les C_j qui sont négatifs
3. Calculer $\Delta_j = Z_j - C_j$
4. Choisir l'indice sortant r tq $b_r = min_{b_i<0}\ b_i$; si b_r n'existe pas alors l'optimum est atteint
5. Choisir l'indice entrant k tq $\frac{\Delta_k}{a_{rk}} = min_{a_{rj}<0} \frac{\Delta_j}{a_{rj}}$, le pivot a_{rk} doit être négatif

Si $a_{rj} \geq 0\ ;\ \forall j$, l'optimum est non borné (les contraintes du dual sont contradictoires).

Exemple

Soit le PL suivant :

$$(P) \begin{cases} min & Z = x_1 + x_2 \\ sc & 2x_1 - x_2 \geq 2 \\ & -x_1 + 2x_2 \geq -2 \\ & x_1 + x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{cases}$$

la forme standard

$$(P) \begin{cases} min & Z = x_1 + x_2 \\ sc & 2x_1 - x_2 - x_3 = 2 \\ & -x_1 + 2x_2 - x_4 = -2 \\ & x_1 + x_2 + x_5 = 5 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

$$(P)\begin{cases} min & Z = x_1 + x_2 \\ sc & -2x_1 + x_2 + x_3 = -2 \\ & x_1 - 2x_2 + x_4 = 2 \\ & x_1 + x_2 + x_5 = 5 \\ & x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

▷ Tous les C_j sont positifs

Tableau 1

C_B	base	b	x_1	x_2	x_3	x_4	x_5
0	$\leftarrow x_3$	-2	-2	1	1	0	0
0	x_4	2	1	-2	0	1	0
0	x_5	5	1	1	0	0	1
			-1↑	-1	0	0	0

Tableau 2

C_B	base	b	x_1	x_2	x_3	x_4	x_5
1	x_1	1	1	$-\frac{1}{2}$	$-\frac{1}{2}$	0	0
0	x_4	1	0	$-\frac{3}{2}$	$\frac{1}{2}$	1	0
0	x_5	4	0	$\frac{3}{2}$	$\frac{1}{2}$	0	1
		Δ_j	0	$-\frac{3}{2}$	$-\frac{1}{2}$	0	0

Les b_i sont tous positifs donc l'optimum est atteint.
La solution optimale est : $x = (1, 0, 0, 1, 4)$

1.10.4 Contrainte artificielle dans la méthode duale du simplexe [9]

Soit le problème linéaire suivant :

$$(P)\begin{cases} min & Z = Cx \\ & Ax = b \\ & x \geq 0 \end{cases}$$

à résoudre par la méthode duale simplexe

▷ Condition d'application de la méthode du duale simplexe $C \geq 0$.

▷ La méthode de la contrainte artificielle est nécessaire dans la situation ou il existe une base initiale, mais qu'elle n'est pas duale admissible.

▷ A partir de problème (P), un nouveau problème (P') est crée artificiellement, les éléments artificiels introduits sont tels que le problème (P') possédera une base duale admissible, il sera donc possible d'appliquer l'algorithme dual simplexe.

De la conclusion obtenue pour le problème (P') (problème impossible ou obtention d'une solution optimale finie) ; il faudra déduire la conclusion relative au problème (P).

▷ 3 cas peuvent se présenter :

1. (P') n'a pas de solution $\Rightarrow$ (P) n'a pas de solution

2. (P') a une solution optimale mais la variable artificielle est absente de la base $\Rightarrow (P)$ a une solution infinie
3. (P') a une solution optimale et la variable artificielle fait partie de la base finale $\Rightarrow (P)$ a une solution optimale

Exemple :

soit le problème suivant :

$$(P)\begin{cases} min \quad Z = -2x_1 - 3x_2 + x_3 \\ -4x_1 - 5x_2 - 2x_3 \geq 5 \\ -x_1 + 6x_2 + x_3 \geq 10 \\ x_1, x_2, x_3 \geq 0 \end{cases}$$

La forme standard

$$(P)\begin{cases} min \quad Z = -2x_1 - 3x_2 + x_3 \\ -4x_1 - 5x_2 - 2x_3 - x_4 = 5 \\ -x_1 + 6x_2 + x_3 - x_5 = 10 \\ x_1, x_2, x_3, x_4, x_5 \geq 0 \end{cases}$$

Soit M une constante symbolique aussi grande que l'on veut.
$C_1 = -2 < 0$; $C_2 = -3 < 0$
$x_1 + x_2 \leq M$
$x_1 + x_2 + x_6 = M$
$x_2 = M - x_1 - x_6$
substituer x_2 dans Z et les contraintes.
Le problème (P) est équivalent à :

$$\begin{cases} min \quad Z = x_1 + x_3 + 3x_6 - 3M \\ x_1 - 2x_3 - x_4 + 5x_6 = 5 + 5M \\ -7x_1 + x_3 - x_5 - 6x_6 = 10 - 6M \\ x_1 + x_2 + x_6 = M \\ x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} min \quad Z = x_1 + x_3 + 3x_6 - 3M \\ -x_1 + 2x_3 + x_4 - 5x_6 = -5 - 5M \\ 7x_1 - x_3 + x_5 + 6x_6 = -10 + 6M \\ x_1 + x_2 + x_6 = M \\ x_1, x_2, x_3, x_4, x_5, x_6 \geq 0 \end{cases}$$

La condition d'application de la méthode duale simplexe est vérifiée $(C \geq 0)$.
On applique la méthode du duale simplexe, dont la base initiale est $x = (x_4, x_5, x_2)$.

Chapitre 2

Présentation des problèmes linéaires mixtes en nombres entiers et méthodes de résolution

2.1 Introduction

Dans la plupart des problèmes que nous traitons en programmation linéaire, les variables considérées sont supposées à valeurs réelles. Dans certaines applications, on peut se trouver dans d'autres situations où toutes les variables doivent être entières (programmation linéaire en nombres entiers PLNE), ou toutes les variables doivent être binaires (programmation en variables binaires PLVB), ou bien certaines variables peuvent être entières ou binaires, les autres continues (programmation linéaires mixtes PLVM).

Optimisation combinatoire

Un problème d'optimisation combinatoire, est un programme mathématique particulier, il consiste à trouver la meilleure solution parmi un nombre fini (mais souvent très grand) d'éléments.

De nombreux problèmes d'optimisation combinatoire issus d'application réelles peuvent être modélisés comme des PLNE particuliers, ainsi un algorithme efficace pour les PLNE, permettant de résoudre beaucoup de problèmes d'optimisation combinatoire.

La complexité algorithmique [8],[17]

Afin de mesurer la difficulté d'un problème donné et la comparer avec celles des autres problèmes, nous pouvons calculer la complexité algorithmique de chacun d'entre eux.

La complexité d'un problème donné est discutée sur deux cotés :

- ▷ Le coté temporel "complexité temporelle".
- ▷ Le coté spatial "complexité spatiale".

Bien que la théorie de la complexité se concentre sur des problèmes de décision, elle peut être étendue aux problèmes d'optimisation, elle classe les problèmes selon leurs complexités en trois classes :

La classe NP

La classe NP contient tous les problèmes dont nous pouvons vérifier qu'une solution donnée est réalisable en temps polynomial.

La classe P

La classe P contient l'ensemble des problèmes polynomiaux.

Un problème est dit polynomial s'il existe un algorithme polynomial pour le résoudre. On considère les problèmes appartenant à cette classe comme étant facile.

Ex : problème Eulérien.

La classe NPC

Un problème est dit NP-complet lorsqu'il est dans la classe NP, et si tout problème de la classe NP peut se réduire polynomiale-ment à lui.

Remarque :

P est une classe des problèmes les plus faciles de NP ; les plus difficiles sont ceux de NPC.

En général, dans les algorithmes se rapportant aux graphes, on montre qu'un problème est NP-complet par comparaison avec un autre problème connu NP-complet, en utilisant la transformation polynomiale.

▷ On peut, alors représenter les différentes classes comme suit :

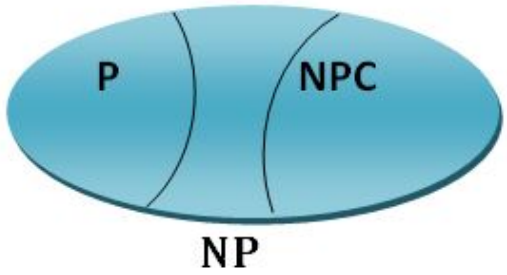


FIGURE 2.1 – Les différentes classes de la complexité

2.2 Formulation du problème [9],[16],[14]

▷ Un programme linéaire en nombres entiers (PLNE) se présente comme suit :

$$(P)\begin{cases} max & Z = C^t x \\ & Ax \geq b \\ & x \in \mathbb{Z}^n \end{cases}$$

$C \in \mathbb{R}^n$; $b \in \mathbb{R}^m$; $A \in \mathbb{R}^{m \times n}$

Si $x_j \in \{0,1\} \quad \forall j = 1,...,n$ on dit qu'on a un programme linéaire en $0,1$ (ou bivalent)

▷ Un programme linéaire mixte en nombres entiers (PLVM) se présente comme suit :

$$(P)\begin{cases} max & Z = C^t x \\ & Ax \geq b \\ & x_j \in \mathbb{R}^+ \quad j = 1,....,n \\ & x_j \in \mathbb{N} \quad j = 1,...,k \quad k \leq n \end{cases}$$

La relaxation linéaire

La relaxation la plus évidente de ce problème consiste à relâcher les contraintes d'intégrité, on obtient un programme linéaire :

$$(PR)\begin{cases} max & C^t x \\ & Ax \geq b \\ & x \in \mathbb{R}^n \end{cases}$$

(PR) le problème relaxé de (P)

Remarque importante

Les programmes linéaires mixtes en nombres entiers se résolvent avec la même technologie que les programmes en nombres entiers (ou vice versa) vue que, la résolution de PLNE ou de PLVM se ramène à la relaxation continue.

Les avantages de la PLNE et PLVM [12]

▷ On peux modéliser plus de problèmes comme PLNE et PLVM que comme PL
▷ Il y a des méthodes pour résoudre les PLNE, qui sont souvent efficaces en pratique
▷ Il y a des solveurs qu'on peut utiliser pour appliquer ces méthodes rapidement

La complexité des problémes PLNE (PLVM) [9],[16]

Un PLNE (ou PLVM) est souvent beaucoup plus difficile a résoudre que sa relaxation continue, il arrive même que ce soit impossible a faire dans un temps raisonnable.

→Le PLNE (PLVM) est NP-difficile.

Malgré que le problème PLNE (ou PLVM) ne contient qu'un nombre fini de solutions admissibles (si le polyèdre S est borné), alors que le problème (PL) contient une infinité non dénombrable de solutions admissibles, le premier problème est beaucoup plus difficile que le second [9].

2.3 Problémes linéaires à solutions entières [11]

Certains problèmes à coefficients entiers ont naturellement des solutions entières.

Définition :

Une matrice totalement unimodulaire est une matrice dont toutes les sous-matrices carrées régulières ont un déterminant égal à $+1$ ou -1.

Exemple : matrice de permutation.

Propriété (condition suffisante) [10]

Si $A \in Z^{m \times n} \quad m \leq n$, est totalement unimodulaire, avec $b \in Z^m$, alors une solution optimale x^* du problème linéaire continu (PL) est entière : $x^* \in Z^n$.

Propriété :

La matrice d'incidence sommets-arrêtes d'un graphe est totalement unimodulaire si et seulement si le graphe est biparti (c'est le cas en particulier pour les problémes d'affectation)

Remarque :

La condition d'unimodulaire est suffisante mais non nécessaire pour l'existence d'une solution optimale entière du problème (PL). En pratique, de nombreux problémes d'optimisation à données entières ont une solution entière bien que la matrice A ne soit pas unimodulaire, mais l'intégrité de la solution risque d'être perdue si des paramètres changent.

2.4 Solution du PL vs solution du PLNE

Propositions [10] :

Soit (P') le problème relaxé de (P) (un PLNE)

- ▷ $Z' \geq Z$ (la valeur du (P') fournit une borne supérieure (ou inf si on minimise) sur la valeur du (P))
- ▷ Si le problème relaxé (P') est irréalisable, le problème original (P) est aussi irréalisable.

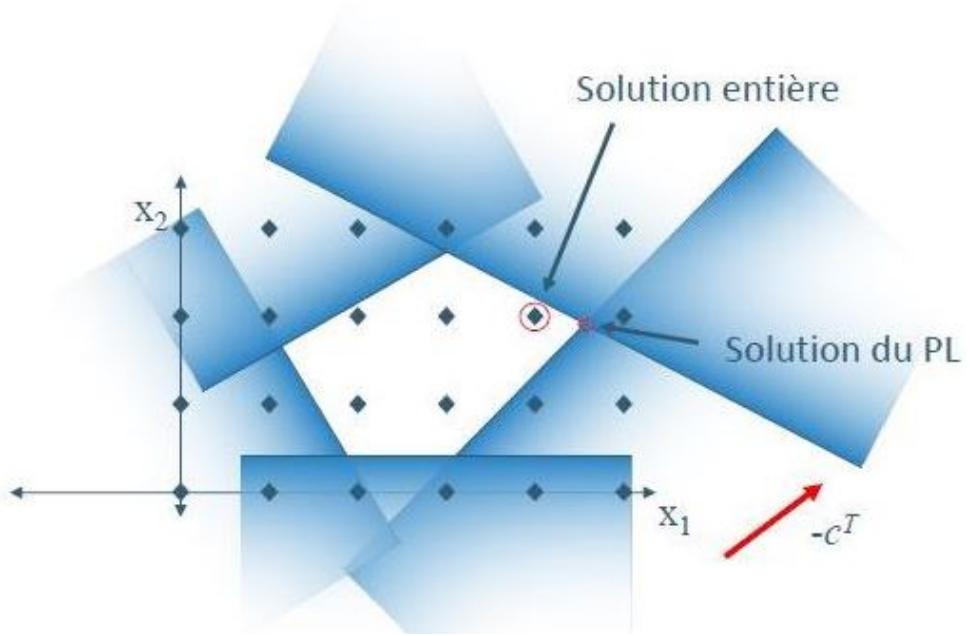


FIGURE 2.2 – Solution du PL vs solution du PLNE

▷ En général, arrondir la solution du PL ne fonctionne pas (voir la figure 2.3) car une solution arrondie peut :
. Ne pas être réalisable pour le PLNE
. Être éloignée de la solution du PLNE

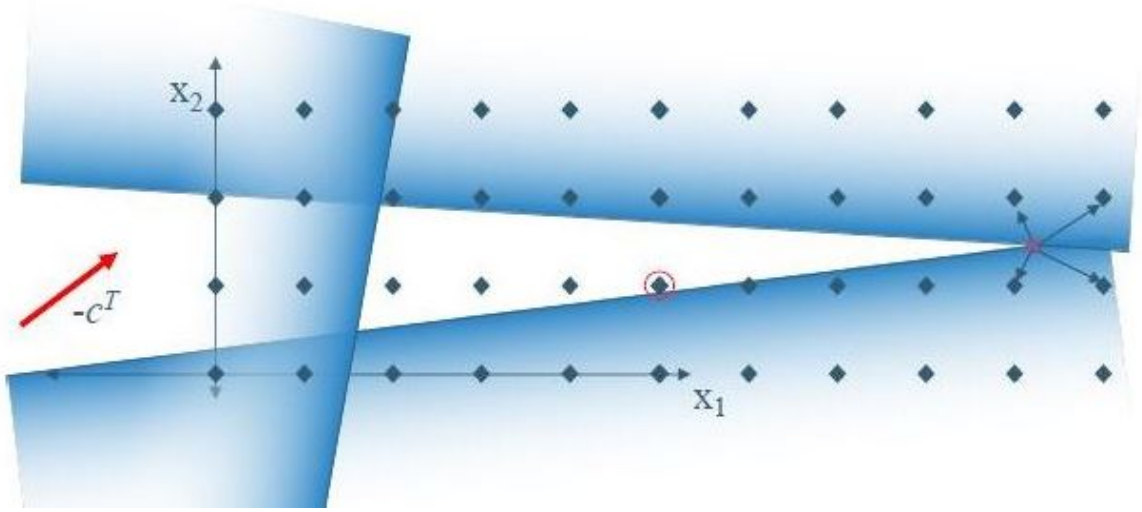


FIGURE 2.3 – Arrondissement des solutions réelles

Méthodes de résolutions

1. Les méthodes exactes

La méthode de résolution exacte doit permettre l'obtention d'au moins une solution optimale. Ce genre de méthode demande en général des temps d'exécution très élevés ainsi que des ressources mémoire importantes sur les instances de grande taille.

Nous citons quelques méthodes de la classe des algorithmes exacts, ces méthodes donnent une garantie de trouver la solution optimale pour une instance de taille finie dans un temps limité et de prouver son optimalité :

- ▷ La méthode de séparation et évaluation (Branch and Bound)
- ▷ La méthode des coupes planes (cutting-plane)
- ▷ La méthode (Branch and Cut)

2. Les méthodes approchées (heuristiques)

En optimisation combinatoire, une heuristique est un algorithme approché qui permet d'identifier en un temps polynomial au moins une solution réalisable rapide, pas obligatoirement optimale. L'usage d'une heuristique est efficace pour calculer une solution approchée d'un problème et ainsi accélérer le processus de résolution. Généralement une heuristique est conçue pour un problème particulier, en s'appuyant sur sa structure propre sans offrir aucune garantie quant à la qualité de la solution calculée.

Comparaison entre les méthodes exactes et les méthodes approchées [13]

1. **Les méthodes exactes**
   - Elles trouvent la solution optimale.
   - Elles peuvent prendre un nombre exponentiel d'itérations.
2. **Les heuristiques**
   - Elles produisent une solution sous-optimale.
   - Elles ne produisent pas de mesure de qualité de la solution.
   - En général, elles ne prennent pas un nombre exponentiel d'itérations.

2.5 Algorithmes de résolutions

Les idées algorithmiques principales

Soit le problème suivant :

$$(P) \begin{cases} min & F(x) \\ & x \in Z^n \end{cases}$$

- ▷ Si on peut dire à priori que $x^* \in Z^n$, alors on peut juste résoudre le (PR) la relaxation de (P) au lieu de (P).
- ▷ On peut ajouter des contraintes à (P) pour obtenir (P') tel que $x^{*'} \in Z^n$ (L'algorithme des coupes).
- ▷ On peut énumérer les solutions de façon intelligente (Branch and Bound).
- ▷ Utiliser les solveurs modernes.

2.5.1 Le Branch and Bound

Schéma général d'une procédure par séparation et évaluation

Les méthodes dites de Branch and Bound **"B&B"** (procédure par séparation et évaluation) s'écartent résolument de ce schéma classique et ont été spécialement élaborées pour des problémes en variables discrètes.

$$(P) \begin{cases} min & F(x) \\ & x \in S \subset Z^n \end{cases}$$

Où S contient un nombre fini de solutions.

La principale originalité des méthodes **B&B** consiste, à chaque étape, à subdiviser l'ensemble S en un nombre fini de sous-ensembles $S^{(i)}$ tels que :

$$\bigcup_{i=1}^{p} S^{(i)} = S$$

De cette façon, le problème se réduit a l'étude des p problèmes $\left(P^{(i)}\right)$ plus restreints :

$$\left(P^{(i)}\right) \begin{cases} min & F(x) \\ & x \in S^{(i)} \end{cases}$$

Il sera généralement conseillé de choisir une subdivision qui correspond a une partition de S c'est à dire qui vérifie également :

$$S^{(i)} \cap S^{(j)} = \emptyset \quad \forall (i,j) \quad i \neq j$$

Ces subdivisions successives sont représentés à l'aide d'une arborescence

- ▷ Le sous ensemble de niveau zéro (la racine de l'arbre) correspond au problème initial (P) avec $S^{(0)} \equiv S$.
- ▷ Les sous ensembles de niveau 1 sont repéré par un seul indice.
- ▷ Chaque sous ensemble de niveau q (> 1) est repéré par q indices dont les $q-1$ premiers indiquent le sous ensemble "parent" dont il est issu.

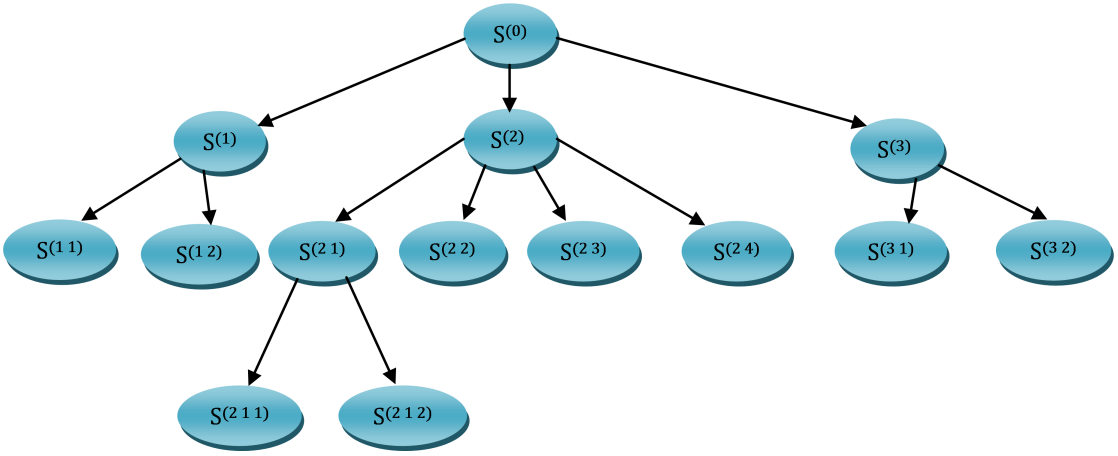


FIGURE 2.4 – Arborescence d'une méthode "Branch and Bound"

▷ Une méthode de **B&B** sera essentiellement constitué de trois éléments principaux :

1. **Une procédure de séparation :** qui à chaque fois qu'il sera nécessaire de séparer un sous ensemble de solutions, déterminera combien de nouveaux sous ensembles considérer et comment les constituer. Comme l'identique la figure précédente, le nombre de sous ensembles créés n'est pas nécessairement identique a chaque séparation ; il peut dépendre du résultat de l'analyse du sous problème parent.
2. **Une procédure d'évaluation :** qui consiste a analyser un sous problème pour l'essentiel, cette analyse vise a évaluer la valeur optimale de la fonction économique du sous problème, plus précisément a déterminer une borne inférieure (supérieure pour un problème de maximisation) de cette valeur.
3. **Une procédure de cheminement :** qui indique quels sous ensembles analyse et dans quels ordre. bien évidement il est souhaitable d'examiner le moins de sous ensembles possibles : certains d'entre eux pourront ne pas être séparer car, par exemple, leur analyse mettra en évidence qu'ils ne contient pas de solutions meilleurs que celles déjà trouvées. Lorsque un nœud de l'arborescence est sondé, il conviendra de remonter dans l'arborescence.

Validité des algorithmes par séparation et évaluation

1. Convergence :

Soit K une famille de sous-ensembles disjoints recouvrant S
ie : $K = \{S_1, S_2, ...S_k\}$ où $S_i \subset S \quad \forall i = 1, ..., k \quad S_i \cap S_j = \emptyset \quad \forall\, i \neq j$ et $\bigcup_{i=1}^{k} S_i = S$.
Il est claire qu'à chaque étape, K représente un recouvrement de S. Il s'ensuit que le critère d'arrêt (procédure de choix) est justifié.
D'où, la garantie de la convergence de la méthode.

2. Finitude :

À cause de la possibilité de fausses séparations, elle doit être prouvée pour chaque algorithme décrit. Alors que pour la finitude, on a, à chaque fois qu'on passe à la subdivision de l'ensemble S, il faudrait que cette subdivision définissent une partition de S.

Un algorithme de Branch and Bound pour la programmation linéaire mixte en nombres entiers

Soient :
x^* : la solution du (P) problème $(PLNE)$
$x^{*'}$: la solution du $\left(P'\right)$ le problème relaxé (PR)
Z^* : la valeur de la fonction économique du (P) problème $(PLNE)$
$Z^{*'}$: la valeur de la fonction économique du (P') le problème relaxé (PR)

1. Résolution du problème relaxé (PR) par la méthode du simplexe :

▷ Si $x^{*'}$ est entier : fin
▷ Si non aller à **2.**

2. Initialisation :

Soit $Z^{*'}$ obtenu dans (P') une borne supérieure pour un problème de maximisation respectivement (borne inférieure pour un problème de minimisation) pour (P).

Soit n_1 le sommet initial de l'arborescence et son ensemble $(S_1 = S)$.

$Z_1 = Z^{*'}$

3. Séparation ($k^{éme}$ itération) :

Choisir une variable non entière x_l, créer deux branches (deux sommets fils n_{i+1} et n_{i+2}), on obtient deux sous-problèmes sous la forme :

$$\begin{cases} P_{i+1,k} : & P_i + \textit{ la contrainte } x_l \leq [x_l] \\ P_{i+2,k} : & P_i + \textit{ lacontrainte } x_l \geq [x_l] + 1 \end{cases}$$

Avec $[x_l]$ la partie entière de x_l (le grand nombre entier inférieur ou égale au nombre x_l)

4. Résolution des sous-problèmes :

Résoudre chaque sous-problème en utilisant le simplexe ou le dual simplexe.

5. Évaluation :

Examiner chaque sous-ensemble :

On peux tailler un sommet si :

- ▷ La solution est non réalisable
- ▷ $Z_1 \leq Z$ pour un problème de maximisation ($Z_1 \geq Z$ pour un problème de minimisation), avec Z la solution du sous-problème.
- ▷ La solution est non entière et son Z inférieur ou égale à une solution entière pour un problème de maximisation (supérieure ou égale pour un problème de minimisation).

Il est inutile de séparer si :

- ▷ La solution est entière

6. Test :

S'il y a plus de sous-ensembles à séparer, alors :

On compare tous les Z des solutions entières et on prend la plus grande d'entre elles soit Z^* pour un problème de maximisation (la plus petite pour un problème de minimisation). Elle sera la valeur de la fonction économique de la solution optimale x^* de notre problème (P).

Sinon retour à **3.**

Remarque :

$Z \leq Z^*$ pour un problème de maximisation ($Z \geq Z^*$ pour un problème de minimisation).

Dans l'itération **3.** pour séparer les sous-problèmes on utilise l'une des stratégies cités précédemment.

Exemple 1 :

Considérons le PLNE suivant :

$$(P_0)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x \in \mathbb{N}^2 \end{cases}$$

▷ Résoudre $\left(P_0'\right)$ le problème linéaire relaxé de (P_0)

$$\left(P_0'\right)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x \geq 0 \end{cases}$$

La résolution (avec la méthode graphique ou Simplexe) donne $x^{*'} = \left(\frac{15}{4}, \frac{15}{16}\right)$, $Z^{*'} = \frac{135}{16}$
Pour S_0 l'ensemble des solutions réalisables de (P_0), considérons la partition :
$S_1 = \{x \in S_0 : x_1 \leq 3\}$
$S_2 = \{x \in S_0 : x_1 \geq 4\}$

$$S_0 = S_1 \cup S_2 \; et \; S_1 \cap S_2 = \emptyset$$

On associe à S_1 et S_2 respectivement les PLNEs suivants :

$$(P_1)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \leq 3 \\ & x \in \mathbb{N}^2 \end{cases}$$

$$(P_2)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \geq 4 \\ & x \in \mathbb{N}^2 \end{cases}$$

Résoudre (P_1') et $\left(P_2'\right)$ les PLs relaxés respectifs de (P_1) et (P_2) :

$$\left(P_1'\right)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \leq 3 \\ & x \geq 0 \end{cases}$$

$$(P_2')\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \geq 4 \\ & x \geq 0 \end{cases}$$

▷ (P_2') n'a pas de solution réalisable car :
$x_1 - 4x_2 \leq 0$ et $x_1 \geq 4$ donc $x_1 - 4x_2 \geq 4 - 4x_2 \Rightarrow x_2 \geq 1 \Rightarrow 3x_1 + 4x_2 \geq 16$ contradiction avec la contrainte $3x_1 + 4x_2 \leq 15$ i.e $S_2 = \emptyset$.

▷ La résolution de (P_1') donne $x^{*(P_1')} = (3, 1.5)$ et $Z^{*(P_1')} = 7.5$
De la même manière, pour S_1, l'ensemble des solutions réalisable de (P_1), considérons la partition :

$S_3 = \{x \in S_1 : x_2 \leq 1\}$ et $S_4 = \{x \in S_1 : x_2 \geq 2\}$

$$S_1 = S_3 \cup S_4 \ et \ S_3 \cap S_4 = \emptyset.$$

On associe a S_3 *et* S_4 respectivement les PLNEs suivants :

$$(P_3)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \leq 3 \\ & x_2 \leq 1 \\ & x \in \mathbb{N}^2 \end{cases}$$

$$(P_4)\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \leq 3 \\ & x_2 \geq 2 \\ & x \in \mathbb{N}^2 \end{cases}$$

Résoudre (P_3') et (P_4') les problèmes relaxés respectivement de (P_3) et (P_4)

$$(P_3')\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \leq 3 \\ & x_2 \leq 1 \\ & x \geq 0 \end{cases}$$

$$(P_4')\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x_1 \leq 3 \\ & x_2 \geq 2 \\ & x \geq 0 \end{cases}$$

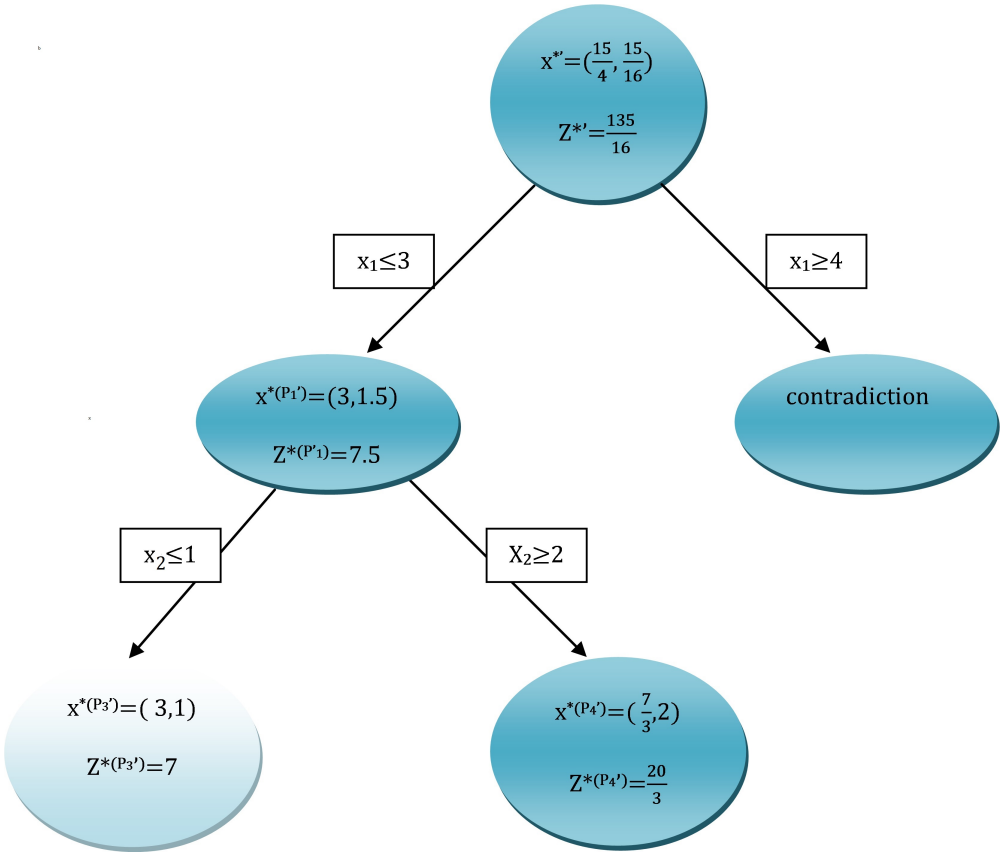
On obtient : $x^{*(P_3')} = (3,1)$; $Z^{*(P_3')} = 7$ et $x^{*(P_4')} = \left(\frac{7}{3}, 2\right)$; $Z^{*(P_4')} = \frac{20}{3}$.

S_3 nous assure que $Z^{*(P_3)} \geq 7$; alors on supprime S_4 car $Z^{*(P_4')} = \frac{20}{3} < 7$.

On s'arrête car la solution optimale de (P_3') est entière. Donc cette solution est optimale pour (P_3).

En conclusion $x^{*(P_3)}$ est optimale pour (P_0) aussi.

En résumé, considérons l'arborescence suivant :



Exemple 2 :

Considérons le PLVM suivat :

$$(P_0)\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_1 \geq 0,\ x_2 \in \mathbb{N} \end{cases}$$

Résoudre (P') le problème relaxé de (P)

$$(P_0')\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_1 \geq 0,\ x_2 \geq 0 \end{cases}$$

▷ La résolution de (P_0') (avec la méthode graphique ou simplexe) donne $x^{*'} = \left(\frac{8}{3}, \frac{10}{3}\right)$; $Z^{*'} = \frac{44}{3}$

Pour S_0 l'ensemble des solutions réalisables de (P_0), considérons la partition :
$S_1 = \{x \in S_0 : x_2 \leq 3\}$
$S_2 = \{x \in S_0 : x_2 \geq 4\}$

$$S_0 = S_1 \cup S_2 \; et \; S_1 \cap S_2 = \emptyset$$

On associe à S_1 et S_2 respectivement les PLVMs suivants :

$$(P_1)\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_2 \leq 3 \\ & x_1 \geq 0,\ x_2 \in \mathbb{N} \end{cases}$$

$$(P_2)\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_2 \geq 4 \\ & x_1 \geq 0,\ x_2 \in \mathbb{N} \end{cases}$$

Résoudre $\left(P_1'\right)$ et $\left(P_2'\right)$ les PLs relaxés respectifs de (P_1) et (P_2) :

$$\left(P_1'\right)\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_2 \leq 3 \\ & x_1 \geq 0,\ x_2 \geq 0 \end{cases}$$

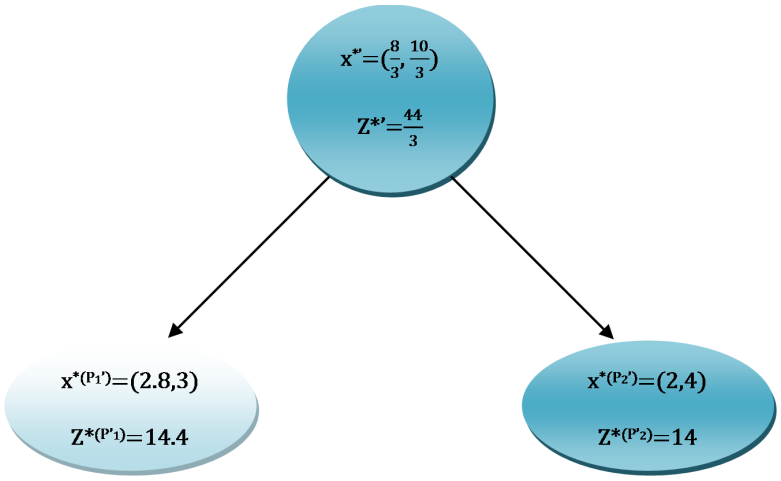
$$\left(P_2'\right)\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_2 \geq 4 \\ & x_1 \geq 0,\ x_2 \geq 0 \end{cases}$$

▷ La résolution de $\left(P_1'\right)$ (par la méthode graphique ou Simplexe) donne $x^{*(P_1')} = (2.8, 3)$ et $Z^{*(P_1')} = 14.4$ (solution admissible)

▷ La résolution de $\left(P_2'\right)$ (par la méthode graphique ou Simplexe) donne $x^{*(P_2')} = (2, 4)$ et $Z^{*(P_2')} = 14$ (solution admissible)

La solution Optimale de (P_0) est $x^* = (2.8, 3)$ et $Z^* = 14.4$

En résumé considérons l'arborescence suivant :



2.5.2 Les méthodes des coupes [14]

Principe des méthodes des coupes [14]

L'idée de base sur laquelle se fondent ces méthodes est la suivante :

On commence par résoudre le programme linéaire continu PL. Si la solution optimale obtenue est entière, l'optimum entier est obtenu et le problème est résolu.

Sinon (si la solution optimale obtenue est non entière), il est facile de voir que l'on peut toujours tronquer le domaine de solution (on rajoute une ou plusieurs contraintes supplémentaires au problème) de façon à éliminer ce point extrême sans exclure aucune solution entière. Une telle contrainte est appelée une coupe.

Beaucoup de coupes peuvent donc être générées pour éliminer une solution fractionnaire donnée.

Après avoir rajouté une coupe ou plusieurs, le programme linéaire augmenté des contraintes correspondantes est à nouveau résolu (en continu) par la méthode du simplexe, il est avantageux d'utiliser pour cela l'algorithme dual simplexe. Si la solution optimale de ce nouveau problème est entière, c'est terminé : on a obtenu une solution optimale de PLNE. Sinon ; le raisonnement précédent peut être répété : on recherchera une nouvelle coupe ou plusieurs... etc

Si les coupes sont correctement choisies à chaque étape, le polyèdre initiale sera ainsi réduit progressivement réduit jusqu'à coïncider avec l'enveloppe convexe des solutions entières et le problème sera résolu.

Il est claire, cependant que les choix des coupes est déterminant pour la convergence de la méthode.

Malheureusement c'est là la difficulté, on ne connaît pas de méthodes systématiques efficaces pour engendrer toutes les équations ou inéquations définissant l'enveloppe convexe des points entiers contenus dans un polyèdre donné. Une des raisons de cette difficulté est qu'elles peuvent être en nombre énorme (résultat de Joroslow 1969 et Rubin 1970), même pour un programme en nombres entiers à deux variables et une contrainte.

C'est pourquoi, un des premiers résultats majeurs en programmation en nombre entiers à été la mise en évidence **(Gomory 1958)** de coupes d'un cas particulier permettant, moyennant certaines précautions, d'obtenir la convergence finie de la méthode vers l'optimum entier.

Algorithme de Gomory pour les PLNEs [4]

On peut résumer la méthode de résolution d'un PLNE (PLVM) avec l'algorithme de Gomory comme suit :

1. Étant donné un PLNE (PLVM), résoudre le PL correspondant à l'aide d'un algorithme du simplexe. Si la solution optimale est entière donc elle est également une solution optimale du PLNE. La résolution est terminée.
2. Si une ou plusieurs variables de base dans la solution optimale du PL ne sont pas entières, on doit alors générer, à partir d'une ligne du tableau une contrainte supplémentaire dite coupe de Gomory (nous indiquons comment construire cette contrainte subséquemment). Cette contrainte (sous forme d'inéquation) est ajoutée au tableau optimal du PL et on détermine le nouveau tableau optimal à l'aide de la méthode duale simplexe.
3. Si les variables de base dans le nouveau tableau optimal sont entières, nous avons obtenu également la solution optimale au PLNE. La résolution est terminée.
4. Sinon, on doit générer (à partir du dernier tableau optimal) une nouvelle coupe de Gomory, et l'ajouter au dernier tableau optimal et trouver la nouvelle solution optimale à l'aide de la méthode duale du simplexe. Si la solution optimale obtenue est à valeurs entières, la résolution est terminée. Sinon, on répète la procédure jusqu'à l'obtention d'une solution optimale à valeurs entières (4.).

La coupe de Gomory

$$\boxed{f_i \leq \sum_{j \in J_H} f_{ij} x_j}$$

Remarque 1 :

On choisira d'abord, dans le tableau optimal, la variable de base qui présente la plus grande partie fractionnaire, et en déduit l'équation (la contrainte) correspondante exprimant toutes les variables du tableau en fonction de la valeur x_B de la variable de base.

Remarque 2 :

Avant d'appliquer l'algorithme de Gomory à la résolution d'un PLNE, introduisons les notions de partie entière et partie fractionnaire d'un nombre.

Un nombre réel a s'écrit comme : $a = [a] + f$.

Ou : $[a]$ est la partie entière (le plus grand nombre entier inférieur ou égale au nombre a), et f la partie fractionnaire de a $(0 < f < 1)$.

Exemple

a	$[a]$	$f = a - [a]$
$\frac{1}{2}$	0	$\frac{1}{2}$
$\frac{-2}{13}$	-1	$\frac{11}{13}$

Exemple :

Soit le PLNE suivant :

$$(P)\begin{cases} max & Z = x_1 + 2x_2 \\ sc & 2x_1 + x_2 \leq 3 \\ & x_2 \leq 2 \\ & x_1, x_2 \in \mathbb{N} \end{cases}$$

Le PL associé au (P) :

$$(P)\begin{cases} max & Z = x_1 + 2x_2 \\ sc & 2x_1 + x_2 \leq 3 \\ & x_2 \leq 2 \\ & x_1, x_2 \geq 0 \end{cases}$$

La forme standard

$$(P)\begin{cases} max & Z = x_1 + 2x_2 \\ sc & 2x_1 + x_2 + x_3 = 3 \\ & x_2 + x_4 = 2 \\ & x_1, x_2, x_3, x_4 \geq 0 \end{cases}$$

En appliquant le simplexe pour le PL :

1^{ére} itération :

		C_j	1	2	0	0
C_B	*base*	b	x_1	x_2	x_3	x_4
0	x_3	3	2	1	1	0
0	$\leftarrow x_4$	2	0	**1**	0	1
		Z_j	-1	-2↑	0	0

2^{éme} itération :

		C_j	1	2	0	0
C_B	*base*	b	x_1	x_2	x_3	x_4
0	$\leftarrow x_3$	1	**2**	0	1	-1
2	x_2	2	0	1	0	1
		Z_j	-1↑	0	0	2

3^{éme} itération :

		C_j	1	2	0	0
C_B	*base*	b	x_1	x_2	x_3	x_4
1	x_1	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$-\frac{1}{2}$
2	x_2	2	0	1	0	1
		Z_j	0	0	$\frac{1}{2}$	$\frac{3}{2}$

La solution optimale $x^* = \left(\frac{1}{2}, 2\right)$ et $Z^* = \frac{9}{2}$

▷ La solution est optimale pour PL mais n'est pas optimale pour PLNE.

Donc on continue avec l'algorithme de Gomory :

$f_i \leq \sum_{j \in J_H} f_{ij} x_j$

$\frac{1}{2} \leq \frac{1}{2}x_3 + \frac{1}{2}x_4 \Rightarrow -\frac{1}{2} \geq -\frac{1}{2}x_3 - \frac{1}{2}x_4 \Rightarrow -\frac{1}{2}x_3 - \frac{1}{2}x_4 + x_5 = -\frac{1}{2}$

▷ Ajoutant la coupe de Gomory au dernier tableau optimal du simplexe :

		C_j	1	2	0	0	0
C_B	*base*	b	x_1	x_2	x_3	x_4	x_5
1	x_1	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$-\frac{1}{2}$	0
2	x_2	2	0	1	0	1	0
0	$\leftarrow x_5$	$-\frac{1}{2}$	0	0	$-\frac{1}{2}$	$-\frac{1}{2}$	1
		Z_j	0	0	$\frac{1}{2}\uparrow$	$\frac{3}{2}$	0

On résout le tableau par la méthode dual simplexe (voir chap1)
On obtient comme dernier tableau de dual simplexe :

		C_j	1	2	0	0	0
C_B	*base*	b	x_1	x_2	x_3	x_4	x_5
1	x_1	0	1	0	0	-1	1
2	x_2	2	0	1	0	1	0
0	x_3	1	0	0	1	1	-2
		Z_j	0	0	0	1	1

La solution optimale $x^* = \begin{pmatrix} 0 \\ 2 \end{pmatrix}$ et $Z^* = 4$
La solution est optimale pour PL et PLNE.

2.5.3 Méthode de Branch and Cut [6]

Il n'est pas toujours possible de résoudre efficacement les problémes NP-difficiles par la méthode des coupes, de même par la méthode de Branch and Bound.

Il est possible de combiner la méthode de Branch and Bound et la méthode des coupes dans un algrithme, on parle alors de Branch and Cut [5].

Pour résoudre un PLNE ou PLVM, l'algorithme de Branch and Cut commence par résoudre la relaxation linéaire de ce problème, puis on applique la méthode des coupes sur la solution trouvée. Si on n'arrive pas a obtenir une solution entière, le problème est divisé en plusieurs sous problémes (le branchement) qui seront résolus de la même façon.

2.6 Problèmes classiques [9],[15]

Quelques exemples pratiques de la programmation linéaire en nombres entiers (PLNE)

1. Le problème de transport [9]

Présentation du problème :

Une marchandise doit être transportée de m magasins de dépôt vers n centres de consommation. Les centres consommateurs demandent respectivement $d_1, d_2, ..., d_n$ unités de cette marchandise. Le centre d'expédition peuvent en fournir $a_1, a_2, ..., a_m$ unités.

Le coût de livraison (de transport) d'une unité du centre (i) au centre (j) est connu et noté par c_{ij}.

La quantité transportée de (i) vers (j) est notée par x_{ij}.

Le problème s'écrit :

$$\begin{cases} min & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} & & \\ & \sum_{j=1}^{n} x_{ij} & \leq a_i & i = 1, ..., m \\ & \sum_{i=1}^{m} x_{ij} & \geq d_j & j = 1, ..., n \\ & x_{ij} \geq 0 & & \end{cases}$$

- ▷ La 1^{ere} contrainte représente le respect des quantités disponibles dans les m centres de dépôt.
- ▷ La 2^{eme} contrainte représente la satisfaction des demandes en les n centres de consommations.
- ▷ Le problème de transport contient mn variables et $m + n$ contraintes.

Le problème de transport peut s'écrire :

$$\begin{cases} min & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} & & \\ & \sum_{j=1}^{n} x_{ij} & = a_i & i = 1, ..., m \\ & \sum_{i=1}^{m} x_{ij} & = d_j & j = 1, ..., n \\ & x_{ij} \geq 0 & & \end{cases}$$

C'est le cas dans l'état de balance ou état de l'équilibre (disponibilité = demande $\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} d_j$).

Lorsque la disponibilité supérieure à la demande ($\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} d_j$), le modèle de programmation est alors :

$$\begin{cases} min & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} & & \\ & \sum_{j=1}^{n} x_{ij} & \leq a_i & i = 1, ..., m \\ & \sum_{i=1}^{m} x_{ij} & = d_j & j = 1, ..., n \\ & x_{ij} \geq 0 & & \end{cases},$$

Lorsque la disponibilité est inférieure à la demande ($\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} d_j$), le modèle de programmation est :

$$\begin{cases} min & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\ & \sum_{j=1}^{n} x_{ij} & = a_i \quad i = 1, ..., m \\ & \sum_{i=1}^{m} x_{ij} & \leq d_j \quad j = 1, ..., n \\ & x_{ij} \geq 0 \end{cases}$$

Généralement les quantités a_i, d_j sont entières, ce qui est implique que les variables x_{ij} le soient également. On obtient donc un probléme linéaire en nombres entiers (PLNE).

▷ Le problème de transport est un **faux problème en nombres entiers,** Il suffit donc d'étudier la relaxation linéaire [9].

Comme il existe de nombreuses méthodes pour la résolution du problème du transport (règle du coin nord-ouest, méthode de Vogel,...)

2. Le problème d'affectation [9]

Le problème d'affectation peut être considéré comme cas particulier du problème de transport, consiste à affecter n ressources (personnes, équipements, localisation,...) à n activités (travaux, activités, services,...), par exemple affecter n individus à n tâches.

Le problème s'écrit :

$$\begin{cases} min & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \\ & \sum_{j=1}^{n} x_{ij} & = 1 \quad i = 1, ..., n \\ & \sum_{i=1}^{n} x_{ij} & = 1 \quad j = 1, ..., n \\ & x_{ij} \in \{0, 1\} \end{cases}$$

Où :

$i = 1, ..., n$ les tâches.

$j = 1, ..., n$ les individus.

c_{ij} correspond au coût associé à l'affectation de la tâche i à l'individu j.

x_{ij} correspond à l'affectation éventuelle de la tâche i à l'individu j telle que :

$$x_{ij} = \begin{cases} 1 & \textit{si la tâche } i \textit{ associe à l'individu } j \\ 0 & \textit{sinon} \end{cases}$$

la tâche i doit être affecté à un et un seul individu.

L'individu j doit être affecté a une et une seule la tâche.

▷ Le problème d'affectation est un **faux problème en nombres entiers** [9]

3. Le problème de sac à dos "KP" (knapsack problem) [9]

On dispose d'un sac à dos de poids P, et de n importants d'objets, chaque objet à une utilité et un poids.

Le problème consiste à choisir les objets qui feront partie du chargement de manière à maximiser la valeur totale des objets choisis sans dépasser la capacité maximale.

Le problème s'écrit :

$$\begin{cases} max & Z = \sum_{i=1}^{n} u_i x_i \\ & \sum_{i=1}^{n} p_i x_i \leq P \\ & x_i \in \{0, 1\} \quad i = 1, ..., n \end{cases}$$

Où :
n : objets $i = 1, ..., n$
u_i : leurs valeurs (utilités)
p_i : poids de l'objet
P : le poids maximal du sac à dos

$$x_i = \begin{cases} 1 & si\ i\ est\ pris\ dans\ le\ sac\ à\ dos \\ 0 & sinon \end{cases}$$

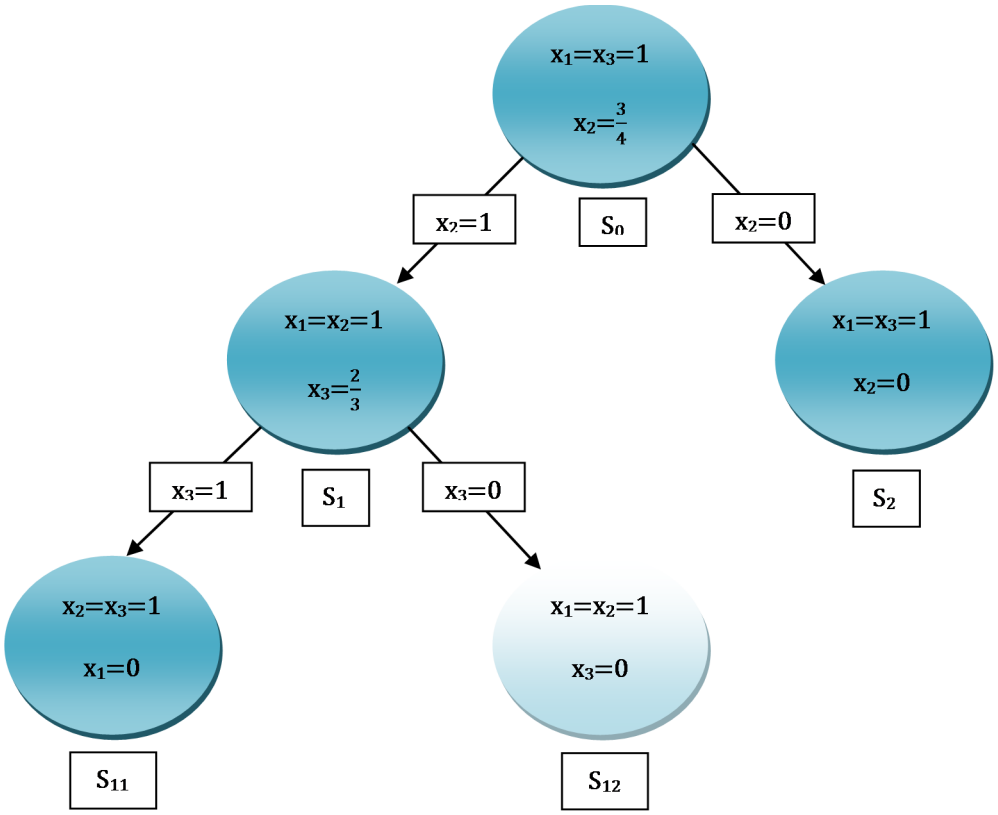
Le problème du sac à dos est un problème en variables {0,1}.
S'il existe plusieurs objets de chaque type ; soit m_i le nombre d'objets de type i.
On obtient donc un problème de sac à dos en variables entières (PLNE), avec $x_i \in \{0, 1, ..., m_i\}$ [9]

Exemple de problème du sac à dos

Soit le problème du sac à dos suivant :

$$\begin{cases} max \quad Z = 16x_1 + 18x_2 + 15x_3 \\ \qquad x_1 + 4x_2 + 3x_3 \leq 7 \\ \qquad\quad x_j \in \{0, 1\} \qquad\quad j = 1, ..., 3 \end{cases}$$

- ▷ L'ordre de préférence : x_1, x_3, x_2
- ▷ L'arborescence associe au problème :



Avec :
$Z_1 = 44$, $Z_2 = 31$, $Z_{11} = 33$, $Z_{12} = 34$

La solution du problème est : $x^* = (1, 1, 0)$, $Z^* = 34$

▷ Le problème du sac à dos, est utilisé pour modéliser divers situations (dans le chargement de bateau ou d'avion, dans la découpe de matériaux, et aussi dans la sélection d'investissement (de manière à maximiser le rendement sans dépasser la somme disponible)...)

4. Le problème de voyageur de commerce "TSP" (Travelling Salesman Problem) [7]

Le problème du voyageur de commerce, est l'un des plus célèbres problèmes de la recherche opérationnelle.

Le problème consiste à déterminer un tour de façon que le voyageur doit visiter chaque ville une et une seule fois (circuit hamiltonien) qui soit de coût minimum.

Le problème s'écrit [9] :

$$\begin{cases} min & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} & \\ & \sum_{j=1}^{n} x_{ij} = 1 & \forall i \\ & \sum_{i=1}^{n} x_{ij} = 1 & \forall j \\ & \sum_{i\in S}\sum_{j\in \overline{S}} x_{ij} \geq 1 & \forall S \\ & x_{ij} = 0, 1 & \end{cases}$$

Où

n : représente les villes

c_{ij} : représente le coût correspondant au l'arc (i, j)

S : représente un sous ensemble de $\{1, ..., n\}$, et $\overline{S}$ son complémentaire

$$\text{et } x_{ij} = \begin{cases} 1 & \textit{si l'arc } (i,j) \textit{ apartient au tour optimal} \\ 0 & \textit{sinon} \end{cases}$$

La solution de TSP doit vérifie deux propriétés :

▷ Le graphe correspondant est connexe.

▷ Chaque sommet est de degré deux (un sommet de ce graphe est une ville).

On peut résoudre le problème de TSP par Branch and Bound, pour cela il faut un moyen de séparer et un moyen d'évaluer, on peut transformer le problème de TSP en un problème d'affectation, le branchement est effectué de manière à éliminer les sous-tours.

Avec :

$$x_{ij} = \begin{cases} 1 & \textit{si le voyage de la ville } i \textit{ à la ville } j \textit{ est effectué} \\ 0 & \textit{sinon} \end{cases}$$

$c_{ii} = M$ élevé (pour imposer $i \neq j$)

La valeur de la solution optimale du problème d'affectation (AP) est une borne inférieure sur la valeur de la solution optimale du TSP.

Si la solution de problème d'affectation est un tour alors elle est également la solution optimale du TSP.

Si un sous tour apparaît dans la solution optimale, alors un de ces arcs doit être absent (pour cela il suffit d'imposer un coût élevé à une des arrêtes de ces sous circuit).

Chacune de ces conditions va correspondre à une branche de l'arbre de Branch and Bound.

Exemple de problème du TSP :

Soit un ensemble de villes $\{ville1,\ ville2,\ ville3,\ ville4,\ ville5\}$, qu'on veut toutes les visiter, au coût moindre, avec la matrice de coût (c_{ij}) suivante :

villes	1	2	3	4	5
1	0	8	2	10	3
2	9	0	20	9	7
3	6	40	0	7	5
4	8	4	2	0	5
5	9	10	6	9	0

On doit imposer c_{ii} élevé.

villes	1	2	3	4	5
1	∞	8	2	10	3
2	9	∞	20	9	7
3	6	40	∞	7	5
4	8	4	2	∞	5
5	9	10	6	9	∞

C'est un problème d'affectation, on peut le résoudre avec la méthode **hongroise** sous une forme qui permet de faire tous les calculs dans la matrice de données du ce problème.

La matrice finale après avoir appliquer toutes les étapes de la méthode hongroise :

villes	1	2	3	4	5
1	∞	3	0	5	0
2	0	∞	14	0	0
3	0	33	∞	0	0
4	5	0	1	∞	3
5	1	1	0	0	∞

Les zéros avec la couleur bleu sont les zéros marqués, les zéros avec la couleur grise sont les zéros éliminés.

Chaque ligne possède une affectation, la solution est donc optimale, et l'algorithme de la méthode hongroise est terminé, et nous avons :

$1 \longrightarrow 5$

$2 \longrightarrow 4$

$3 \longrightarrow 1$

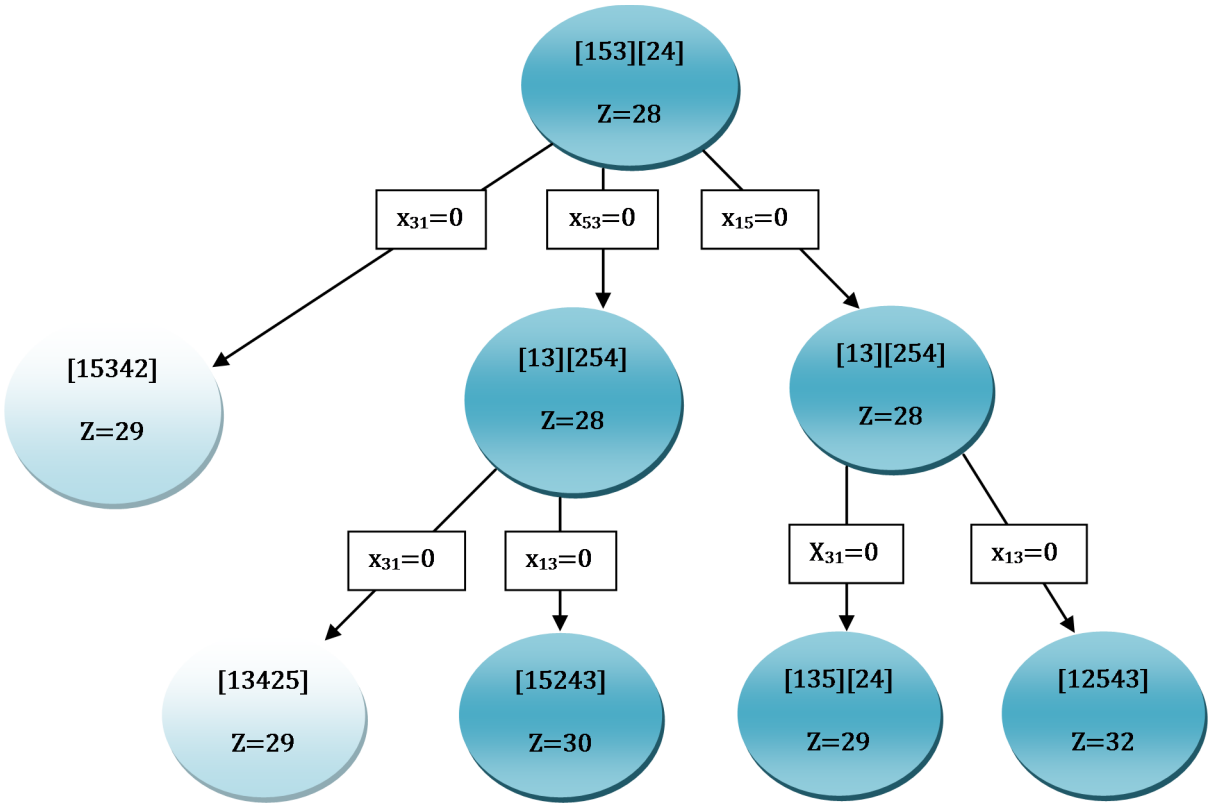
$4 \longrightarrow 2$

$5 \longrightarrow 3$

$V_{aff}(min) = 28$

La solution de problème d'affectation est : $x_{15} = x_{24} = x_{31} = x_{42} = x_{53} = 1$

On a deux sous-circuits [153][24], on peut décider d'éliminer un d'eux (par Branch and Bound).



[13425] et [15342] les solutions de problème TSP, pour un coût $Z = 29$

▷ Le problème de TSP à des applications dans les transports, les reseaux et la logistique,...

Il existe plusieurs d'autre problémes classiques de la programmation linéaire en nombres entiers (problème de recouvrement, problème de partition, problème d'emballage, problème de découpe,...).

Quelques exemples pratiques de la programmation linéaire mixte en nombres entiers (PLVM) [3]

1. Le problème d'implantation de dépôts (avec contraintes de capacités) "Facility Location Problem" [9]

Étant donné un ensemble de sites potentiels pour l'implantation de dépôts, et un ensemble de clients devant être servis par les dépôts installés,

Le problème consiste a déterminer :

1. quels dépôts installer ?
2. comment affecter les clients aux dépôts ?

Le problème s'écrit [9] :

$$\begin{cases} max & \sum_{i\in I}\sum_{j\in J} g_{ij}y_{ij} - \sum_{j\in J} f_j x_j & \\ & \sum_{j\in J} y_{ij} = d_i & \forall i \in I \\ & \sum_{i\in I} y_{ij} \leq q_j x_j & \forall j \in J \end{cases}$$

Où :

I : l'ensemble de clients, $i \in I$.

J : l'ensemble de sites potentiels, $j \in J$.

g_{ij} : représente le gain unitaire correspondant à l'approvisionnement du client i par le dépôt j.

f_j : représente le coût fixe d'ouverture du dépôt j.

$$x_j = \begin{cases} 1 & \textit{si le dépôt } j \textit{ est ouvert} \\ 0 & \textit{sinon} \end{cases}$$

y_{ij} : représente la quantité fournie au client i à partir de dépôt j.

d_i : représente la demande de client i.

q_j : représente la capacité maximale de dépôt j.

▷ Le problème d'implantation de dépôts possède une multitude d'application (implantation des concentrateurs, gestion des aéroports,...).

2. Le problème de coût fixe

Soit un objectif à minimiser :

$$Min\ C^t x$$

Le problème consiste à ajouter un coût fixe F lorsque $x > 0$.

le nouveau problème est donc :

$$\begin{cases} min & Cx + Fy \\ & x \leq My \\ & x \geq 0;\ y \in \{0, 1\} \end{cases}$$

Où :

M la borne supérieure sur x.

ainsi :

si $x > 0$ alors $y = 1$ et on paiera le coût fixe. Sinon, y sera automatiquement mise a 0 pour ne pas payer le coût fixe.

Exemple de problème avec coût fixe :

3 compagnies de téléphone offrent des tarifs différents pour les communications longue distance.

compagnie	abonnement	prix/minute
1	16	0.25
2	25	0.21
3	18	0.22

On veut trouver le plan d'abonnement optimal pour 200 minutes de communication par mois.

Le modèle mathématique de problème :

$$\begin{cases} min \quad Z = 0.25x_1 + 0.21x_2 + 0.22x_3 + 16y_1 + 25y_2 + 18y_3 \\ sc \qquad\qquad x_1 + x_2 + x_3 = 200 \\ \qquad\qquad x_1 \leq 200y_1 \\ \qquad\qquad x_2 \leq 200y_2 \\ \qquad\qquad x_3 \leq 200y_3 \\ \qquad\qquad x_1, x_2, x_3 \geq 0 \\ \qquad\qquad y_1, y_2, y_3 = \{0, 1\} \end{cases}$$

La relaxation linéaire de ce problème est :

$$\begin{cases} min \quad Z = 0.25x_1 + 0.21x_2 + 0.22x_3 + 16y_1 + 25y_2 + 18y_3 \\ sc \qquad\qquad x_1 + x_2 + x_3 = 200 \\ \qquad\qquad x_1 \leq 200y_1 \\ \qquad\qquad x_2 \leq 200y_2 \\ \qquad\qquad x_3 \leq 200y_3 \\ \qquad\qquad 0 \leq y_1 \leq 1 \\ \qquad\qquad 0 \leq y_2 \leq 1 \\ \qquad\qquad 0 \leq y_3 \leq 1 \\ \qquad\qquad x_1, x_2, x_3 \geq 0 \end{cases}$$

La résolution avec Simplexe donne : $(x_1, x_2, x_3, y_1, y_2, y_3)^* = (0, 0, 200, 0, 0, 1)$, $Z^* = 62$

3. Sélection de k contraintes parmi m

Soit un système d'inégalités :

$$\begin{cases} Ax \leq b \\ A \in \mathbb{R}^{m \times n}; \quad x, b \in \mathbb{R}^n \end{cases}$$

▷ On doit satisfaire k contraintes parmi les m contraintes.
Pour cela, on introduit m variables binaires y_i ; $\quad i = 1, ..., m$

On aura donc :

$$\begin{cases} Ax \leq b + My \\ \sum_{i=1}^{m} y_i = k \\ x \in \mathbb{R}^n; \qquad\qquad y \in \{0, 1\}^m \end{cases}$$

Avec :

$$y_i = \begin{cases} 1 & si\ la\ i^{éme}\ contrainte\ est\ satisfaite \\ 0 & sinon \end{cases}$$

$M \in \mathbb{R}$ suffisamment grand

Chapitre 3

Simulations numériques sur des problèmes pratiques

3.1 Introduction

On se trouve souvent dans les situations qui sont trop complexes qu'un être humain puisse considérer tous les aspects importants de la situation, on a besoin de résoudre des modèles avec un montant de données réalistes, c'est à dire trop grand pour trouver une solution réalisable de bonne qualité sans aide informatique.

La programmation est un ensemble d'outils et de technique permettant de résoudre des problèmes mathématiques par ordinateur, elle sert à trouver une solution optimale de n'importe quel type de problème. Le processus de résoudre un problème mathématique exige un grand nombre de calculs donc il vaut mieux l'exécuter sur machine. Pour cela on a choisit le logiciel **LINGO**.

3.2 Présentation de logiciel LINGO

LINGO est un logiciel utilisé pour résoudre les modèles d'optimisation linéaire, non linéaire, et entier...

Une des caractéristiques de **LINGO** c'est qu'il offre des outilles qui peuvent aider à l'analyse des modèles en utilisant la méthode du simplexe.

Les fonctions utilisées dans un modèle de LINGO

@FOR – utilisée pour produira des contraintes
@SUM – calcul de la somme
@MAX – recherche de maximum
@MIN – recherche de minimum

Types des variables dans LINGO

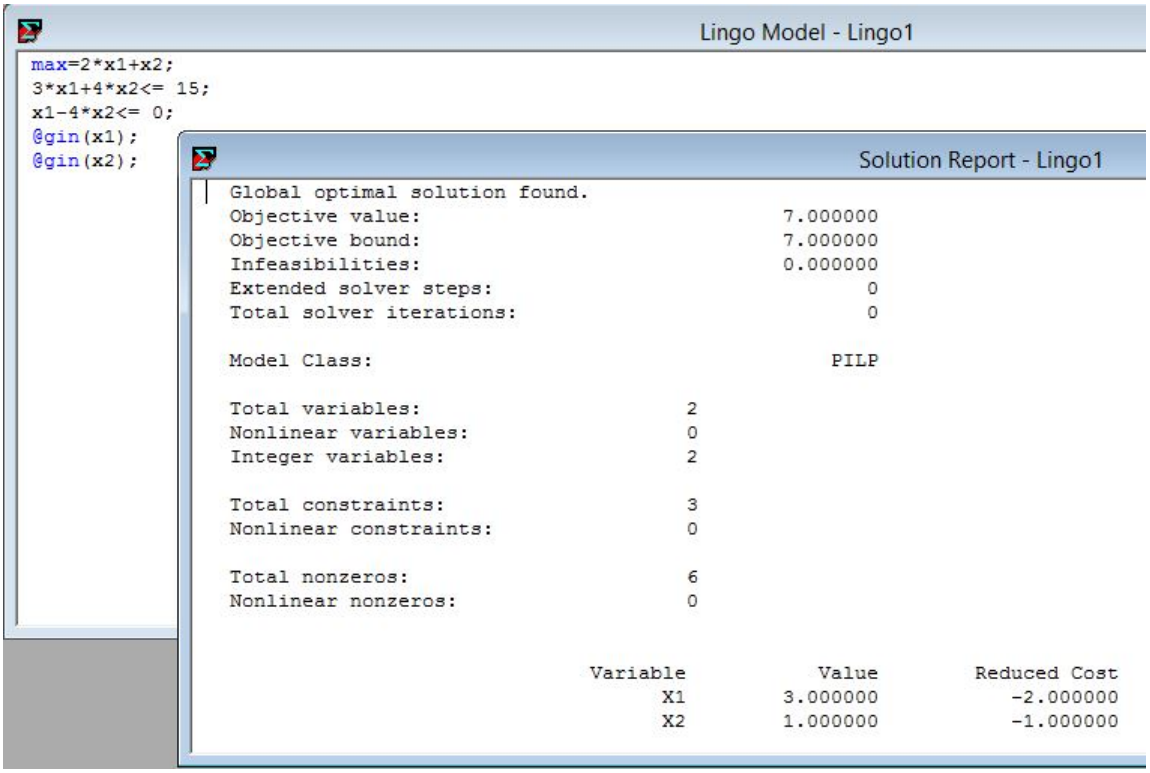
@GIN – toute valeur positive de nombre entier.
@BIN – une valeur binaire (0 ou 1).
@FREE – toute valeur positive ou négative réelle.
@BND – toute valeur bornée par des limites indiquées.

3.3 Résolution de quelques exemples traités dans le deuxième chapitre sur LINGO

3.3.1 Application 1

Soit le problème suivant :

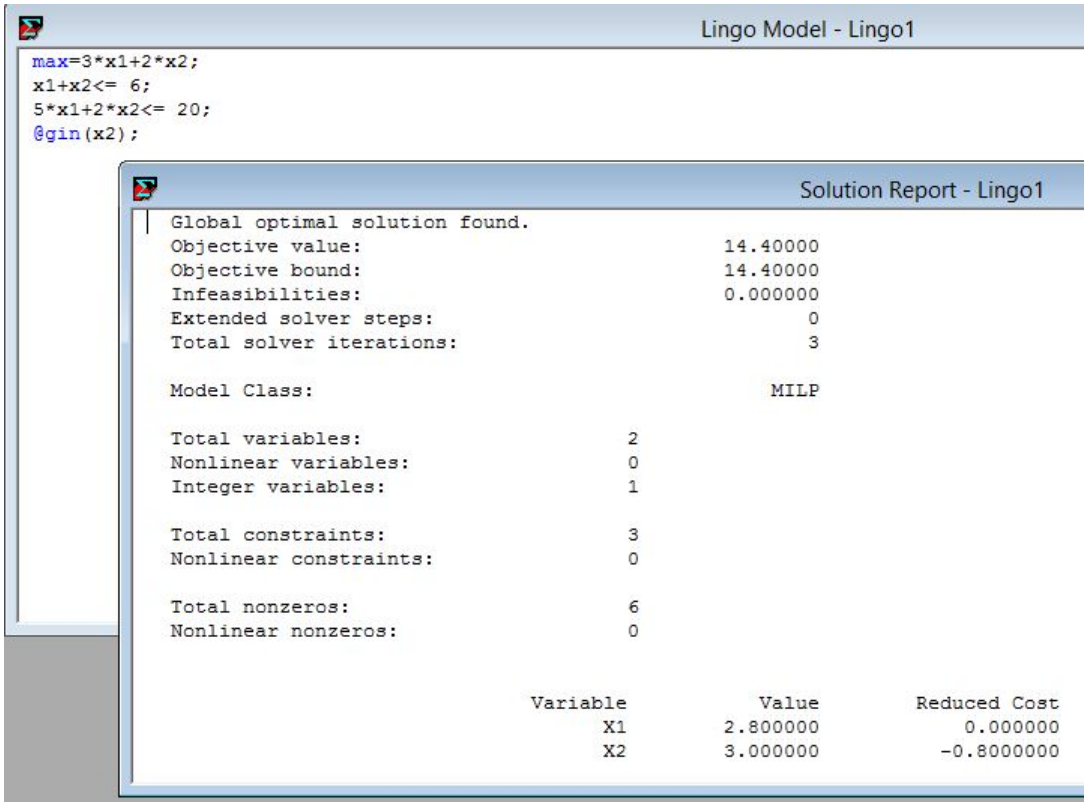
$$\begin{cases} max & Z = 2x_1 + x_2 \\ sc & 3x_1 + 4x_2 \leq 15 \\ & x_1 - 4x_2 \leq 0 \\ & x \in \mathbb{N}^2 \end{cases}$$



3.3.2 Application 2

Soit le problème suivant :

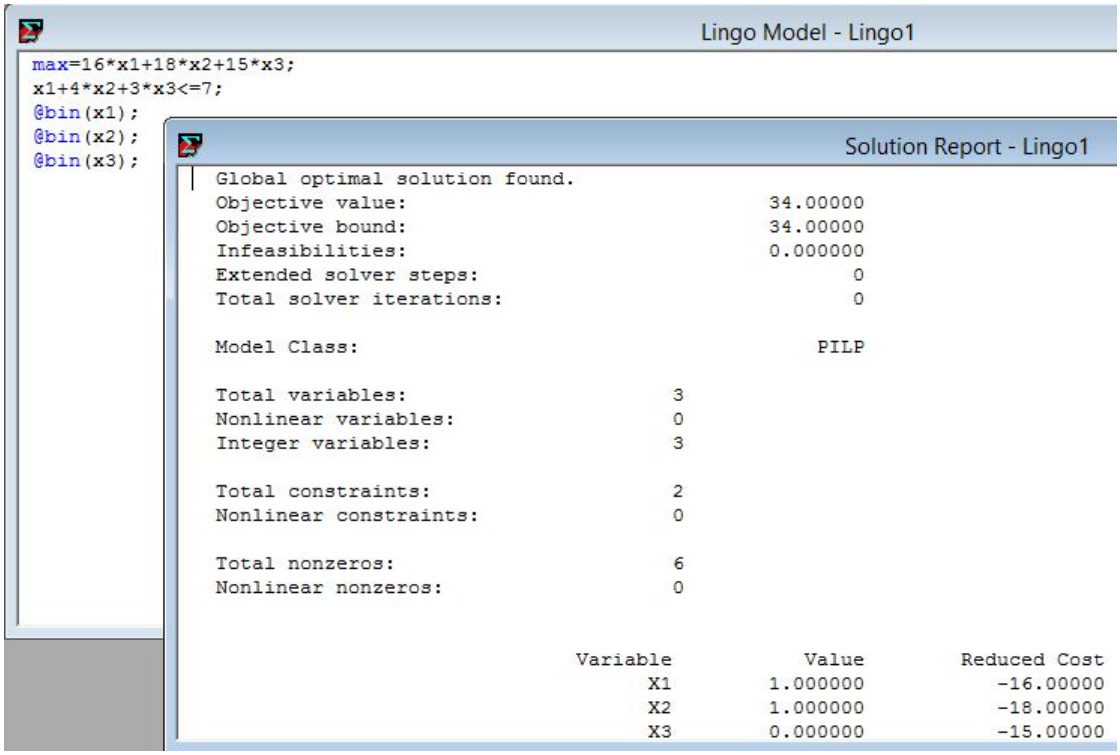
$$\begin{cases} max & Z = 3x_1 + 2x_2 \\ sc & x_1 + x_2 \leq 6 \\ & 5x_1 + 2x_2 \leq 20 \\ & x_1 \geq 0,\ x_2 \in \mathbb{N} \end{cases}$$



3.3.3 Application 3

Soit le problème suivant :

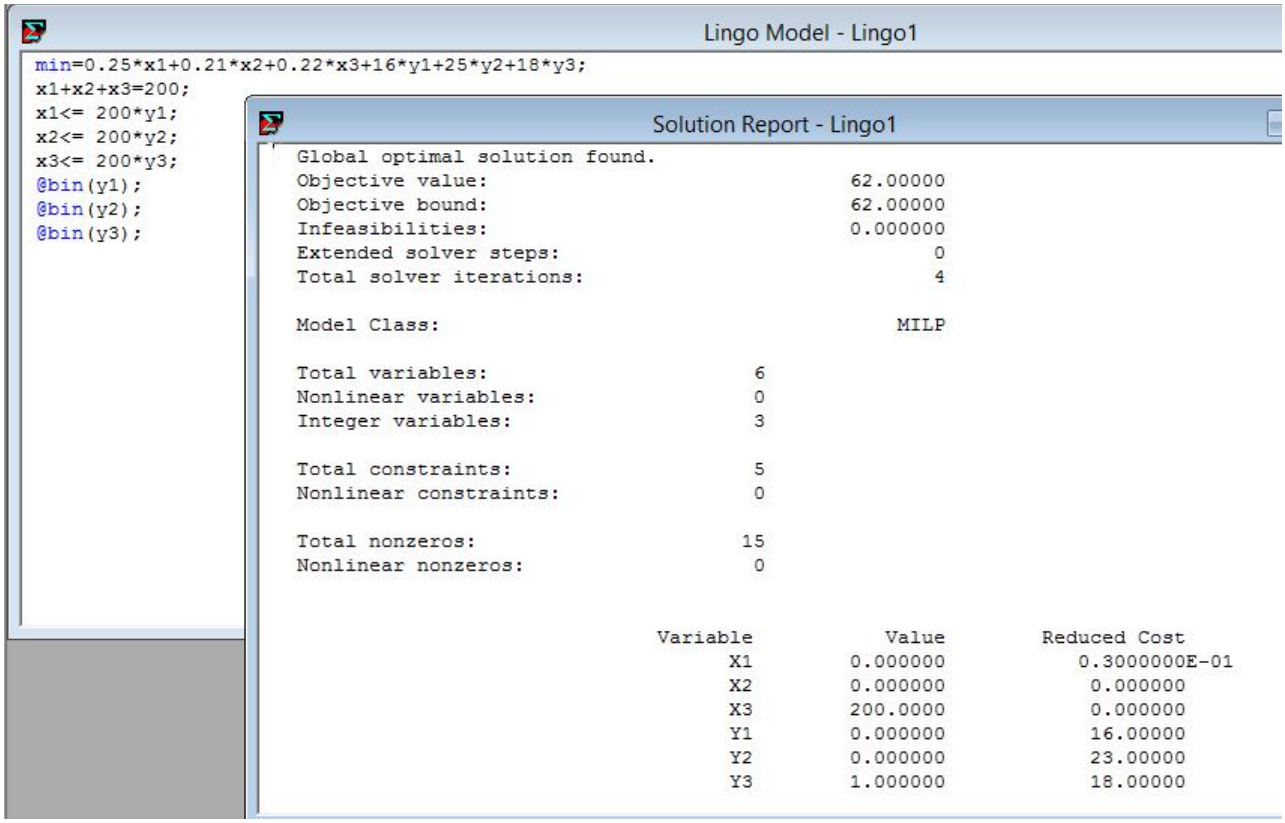
$$\begin{cases} max \quad Z = 16x_1 + 18x_2 + 15x_3 \\ \qquad x_1 + 4x_2 + 3x_3 \leq 7 \\ \qquad\quad x_j \in \{0,1\} \qquad\quad j = 1, ..., 3 \end{cases}$$



3.3.4 Application 4

Soit le problème suivant :

$$\begin{cases} min \quad Z = 0.25x_1 + 0.21x_2 + 0.22x_3 + 16y_1 + 25y_2 + 18y_3 \\ sc \qquad x_1 + x_2 + x_3 = 200 \\ \qquad\qquad x_1 \leq 200y_1 \\ \qquad\qquad x_2 \leq 200y_2 \\ \qquad\qquad x_3 \leq 200y_3 \\ \qquad\qquad x_1, x_2, x_3 \geq 0 \\ \qquad\qquad y_1, y_2, y_3 = \{0, 1\} \end{cases}$$



Conclusion générale

Le problème de la programmation linéaire mixte en nombres entiers est connu pour être central de l'optimisation combinatoire, et l'un des domaines les plus riche de la programmation mathématique, il permet une traduction simple de nombreux problèmes de décision de la classe NPC, ce caractère explique l'importance des travaux et les efforts qui lui on été consacrés.

Après avoir rappelé et présenté certains notions et fondements théoriques de la programmation linéaire, notre travail à consisté en la présentation des programmes linéaires mixtes en nombres entiers, et les méthodes exactes de résolution.

Nous avons présenté les principales méthodes de résolution de ce type de problèmes tels que les méthodes des coupes, les méthodes arborescentes.

Les méthodes arborescentes sont largement utilisées pour résoudre avec succès certains problèmes classiques de la programmation linéaire mixte en nombres entiers.

Finalement, pour montrer l'efficacité des algorithmes présentés dans le deuxième chapitre, on'a traité et vérifié quelques exemples numériques sur le logiciel LINGO.

Ainsi durant ces dernières années, les chercheurs essaient de développer des outils algorithmiques à l'aide des méthodes de séparation et évaluation, et des techniques des coupes, pour remplacer les modèles convexes par des modèles non convexes, mais plus fiable dans les milieux industriels, et élargir ce domaine et les méthodes utilisées, au domaine de la programmation non convexe.

Bibliographie

[1] Aiden M. Oukacha B. "les manuels de l'étudiant Recherche Opérationnelle, Programmation Linéaire". Copyright Eurl Pages Blews internationales, Maison d'Edittion pour l'enseignement et la formation, 2005.

[2] E. Duchêne. Programmation Linéaire, Recherche Opérationnelle, cours de M1, Laboratoire LIRIS. Université Lyon 1.

[3] F. Bastin. Modèles De Recherche Opérationnelle. Université de Montréal, 2010.

[4] G. Baillargeon. Programmation Linéaire Appliquée, Outils d'Optimisation et d'Aide à la Décision. SMG, Québec, 1996.

[5] G. L Nemhauser, L.A Wolsey. Integer and combinational optimization. T. Wiliey, New York, 1988.

[6] H. Boussouira. Méthode de Support pour la résolution des Problèmes Linéaires en Nombres Entiers et Mixtes, Mémoire Magistère, Béjaia, 2009.

[7] H. Talbot. Programmation linéaire en nombres entiers, Laboratoire A2SI, mars 2009.

[8] J.C. Fournier. Théorie des graphes et applications, Lavoisier, 2006.

[9] J. Teghem. Programmation linéaire. Éditions Ellipse 1996.

[10] L. A. Wolsey. Integer Programming. Wiley. Interscience, New Jersy, 1998.

[11] L. A. Wolsey. Valid Inequalities for Mixed Integer Programs. Operations Research, 1976.

[12] L. Liberti, R. Sadykov. Introduction à la Programmation Linéaire en Nombres Entiers. LIX, École Polytechnique, 2006.

[13] M. Bierlaire. Optimisation en nombres entiers. EPFL. Laboratoire Transport et Mobilité, ENAC, 2010.

[14] M. Minoux. Programmation Mathématique Théorie et Algorithme. Tom 2, 1983.

[15] Roseaux. Exercices et problèmes résolus de recherche opérationnelle, Paris 1998.

[16] S. Le Digabel. Optimisation en nombres entiers, Polytechnique, Montréal, H 2020.

[17] S. Vialette. Introduction à la théorie de la complexité. Université Paris-Est Marne La Vallée, 2012.