Titre: Title:	Optimisation d'une fonction non linéaire en variables binaires à l'aide de l'algorithme de Picard-Queyranne
Auteur: Author:	Jean-Bosco Njiyobiri
Date:	1996
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Njiyobiri, J.-B. (1996). Optimisation d'une fonction non linéaire en variables binaires à l'aide de l'algorithme de Picard-Queyranne [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/9003/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/9003/
Directeurs de recherche: Advisors:	Benjamin T. Smith, & Jean-Claude Warmoes
Programme: Program:	Unspecified

UNIVERSITÉ DE MONTRÉAL

OPTIMISATION D'UNE FONCTION NON LINÉAIRE EN VARIABLES BINAIRES À L'AIDE DE L'ALGORITHME DE PICARD - QUEYRANNE

JEAN-BOSCO NJIYOBIRI
DÉPARTEMENT DE MATHÉMATIQUES
ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(MATHÉMATIQUES APPLIQUÉES)
DÉCEMBRE 1996

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

OPTIMISATION D'UNE FONCTION NON LINÉAIRE EN VARIABLES BINAIRES À L'AIDE DE L'ALGORITHME DE PICARD - QUEYRANNE

présenté par: NJIYOBIRI Jean-Bosco

en vue de l'obtention du diplôme: Maîtrise ès Sciences Appliquées

a été dûment accepté par le jury d'examen constitué de :

M. SAVARD Gilles, Ph.D., président

M. SMITH Benjamin T., Ph.D., membre et directeur de recherche

M. WARMOES Jean-Claude, Ph.D., membre et codirecteur

M. PICARD Jean-Claude, Ph.D., membre

A *mon père*, MBONABUCA Simon (+) que je n'ai pas vu quand Dieu l'a rappelé à lui.

A *ma mère* qui a su être toujours présente.

A *ma chère épouse*, Spès, qui a su toujours me supporter durant ces études difficiles.

A *mes enfants*:
Vanessa MUNEZERO, J-Christus NJIYOBIRI et Anniella GATEKA.

Je *vous aime* et *je vous dédie ce mémoire* qui est le fruit de mes efforts et des vôtres.

REMERCIEMENTS

J'aimerais exprimer ma profonde reconnaissance à tous ceux qui, de près ou de loin, m'ont permis de mener à bien ce travail, par leurs conseils, leurs encouragements ou tout autre geste.

Je remercie très sincèrement particulièrement mon directeur de recherche, Monsieur Benjamin T.SMITH qui, grâce à la disponibilité, la détermination et l'encadrement parfait qu'il m'a réservés, ce travail a vu le jour. Je lui dois ici beaucoup de reconnaissance.

J'aimerais exprimer ma reconnaissance à Monsieur Jean-Claude WARMOES, codirecteur, aux encouragements et conseils qu'il m'a toujours donnés.

Je tiens à remercier aussi Monsieur Jean-Claude PICARD qui, en plus d'être le père même de la méthode qui a été developpée dans ce travail, a eu la gentillesse de m'accorder quelques entretiens utiles.

Je pense aussi à Nicolas BÉLANGER, étudiant à la maîtrise au département de mathématiques et génie industriel, qui, malgré ses activités académiques chargées me donnait assez de temps pour discuter de mon sujet surtout sur la programmation. Pour cela, je le remercie vivement.

Je n'oublierais pas Monsieur Beyime TACHEFINE qui fait des recherches sur le problème de la planification de l'exploitation d'une mine à ciel ouvert et qui m'a proposé un algorithme de flot maximum qui fonctionnait correctement.

Je tiens à exprimer ma reconnaissance au personnel technique du laboratoire du CRT (Centre de Recherches sur les Transports, Université de Montréal), aux chercheurs et aux stagiaires pour leurs services courtois pendant les moments les plus durs de mes recherches.

Mes remerciements sont adressés enfin aux membres du Jury qui ont accepté de lire et évaluer ce travail.

RÉSUMÉ

Ce mémoire propose un algorithme exact pour la résolution du problème de maximisation d'une fonction non linéaire sans contrainte, en variables binaires. C'est un problème qui trouve de nombreuses applications réelles surtout dans le cas quadratique. Après avoir démontré qu'une classe de problèmes représentant un cas particulier du problème général peut se résoudre en temps polynômial, en résolvant une relaxation linéaire, on étend cette relaxation au cas général que l'on résout par la technique de séparation et évaluation progressives (S.E.P.). Certaines parties de la S.E.P. ont été programmées, nous permettant ainsi de constater le comportement de l'algorithme en terme de nombre de nœuds dans l'arbre de la S.E.P. nécessaire pour trouver l'optimalité. Nous terminons avec quelques conclusions et recommandations pour des travaux futurs.

ABSTRACT

This thesis proposes an exact algorithm for the solution of the problem of maximizing a nonlinear and unconstrained polynomial function in binary variables. This is a problem with numerous real applications especially in the quadratic case. After having shown that a class of problems representing a special case of the general problem can be solved in polynomial time by solving a linear relaxation, we extend this relaxation to the general case which we solve by the Branch-and-Bound (B & B) technique. Some parts of the B.B. technique have been programmed permitting us to observe the behavior of the algorithm in terms of the number of nodes in the B.B. tree necessary to prove optimality. We finish with several conclusions and recommendations for future works.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SYMBOLES

Ω : l'ensemble des nœuds du graphe qui est aussi un sous-ensemble de $\wp(U)$.

Ω' : l'ensemble des éléments de l'ensemble S appartenant à Ω tels que S contient au moins deux ensembles.

A : l'ensemble des arcs dans le graphe G.

a_S : les pondérations des nœuds du graphe G.

B^n : l'ensemble des nombres binaires.

C_i : une clause qui est une conjonction de littéraux y_j dans la satisfiabilité logique.

C : une constante.

(c_i) : la pondération du nœud i dans le graphe dirigé $G = (N, A)$.

F : une formule logique.

$f(X, Y)$: la fonction des ensembles de variables de décision x et y, ou la fonction relaxée.

$F(X, Y)$: la valeur de la fonction à chaque nœud de calcul.

G : graphe orienté associé à une fonction $f'(X)$.

(i, j) : un arc du réseau.

(k) : est le degré des termes non linéaires.

(l) : est le nombre moyen de variables dans un terme non linéaire de la fonction $f(X)$.

N_1 : est un ensemble de nœuds dont les pondérations sont positives dans le graphe biparti.

N_2 : est un ensemble de nœuds dont les pondérations sont négatives dans le graphe biparti.

(P) : désigne le problème que nous allons résoudre au cours de ce travail.

$(P1)$: le problème (cas particulier) équivalent à P.

$(P2)$: désigne le problème général relaxé.

SEP : désigne la Séparation et Évaluation Progressives.

s : la "source" dans le réseau associé au graphe G.

t : le "puits" dans le réseau associé au graphe G.

U : tous les sous-ensembles de la collection $\wp$.

Y : la fermeture dans le graphe G.

Z : l'ensemble des nombres entiers.

$\wp(U)$: la collection de tous les sous-ensembles de l'ensemble U.

$\emptyset$: un ensemble vide.

CHAPITRE 1
INTRODUCTION

1.1 Énoncé du problème

Le but de ce mémoire est d'élaborer un algorithme pour la maximisation d'une fonction non linéaire polynômiale $f(X)$.

$$\text{Pour } X \in B^n = \{X = (x_1, x_2, ..., x_n) : x_i \in \{0, 1\}, i = 1, 2, ..., n\},$$

on écrit ce problème de la manière suivante:

$$(\text{P}): \quad \max f(X) = \sum_{S \in \Omega} a_S \prod_{i \in S} x_i \tag{1.1}$$

tel que

$$X \in \mathbf{B^n} \tag{1.2}$$

où les pondérations $a_S \in Z$, l'ensemble des nombres entiers, et

$$\Omega \subseteq \wp(\{1, 2, \ldots, n\}) \setminus \emptyset; \tag{1.3}$$

où $\wp(U)$, est la collection de tous les sous-ensembles de l'ensemble U.

L'organisation de ce mémoire est la suivante: nous présentons d'abord une revue de la littérature sur le sujet et nous énonçons un cas particulier de (P) qui se résout en temps polynômial.

Dans le deuxième chapitre, nous exposons les outils dont on aura besoin: la fermeture maximale dans un graphe et le flot maximum et nous résolvons le cas

particulier.

Dans le troisième chapitre, nous revenons au problème général et nous donnons la description de l'algorithme de résolution et un exemple d'illustration de la méthode.

Ensuite, dans le quatrième chapitre nous présentons les résultats numériques de certains problèmes tests tirés de la littérature et engendrés aléatoirement.

Enfin le dernier chapitre contient les conclusions et les recommandations.

1.2 Revue de la littérature

La généralité du problème (P) est due au résultat que n'importe quelle fonction booléenne $f : B^n \longrightarrow Z$ peut être écrite comme un polynôme impliquant les mêmes variables $0-1$. (Gaspar [12], p.21).

De nombreux exemples d'applications de la formulation (P) à des problèmes pratiques ont été cités dans Hansen [14]. Il y cite des applications à des problèmes dans le domaine de la planification des investissements, de la localisation, de l'analyse des "clusters", de la statistique et des graphes et hypergraphes.

La plupart de ces applications utilisent une formulation quadratique $(\{S \in \Omega, |S| = 1 \text{ ou } 2\})$, mais parmi les applications du modèle général, il y a celle du problème de **satisfiabilité logique.**

Une **formule logique F en forme normale disjonctive est une disjonction de "clauses"**, $C_i, i = 1, ..., m$, où chaque clause est une **conjonction** de **littéraux**, des variables binaires pouvant assurer les valeurs **vrai** $= 1$ ou **faux**$= 0$.

Donc F peut s'écrire:

$$F = \bigvee_{i=1}^{n} C_i$$

où

$$C_i = \bigwedge_{j \in A_i} y_j, \qquad A_i \in \Omega \subseteq \wp(1, 2, ..., n) \setminus \emptyset$$

et $\quad y_j = x_j$ ou $y_j = 1 - x_j$

x_j est vrai si et seulement si $1 - x_j$ est faux, $i = 1, ..., n$.

Une clause C_i de F est **satisfaite** si et seulement si la valeur d'au moins l'un de ses littéraux est faux et l'expression booléenne F est satisfaite si et seulement si il existe une affectation de valeurs $vrai$ ou $faux$ aux $x_i, i = 1, ..., n$ telle que toutes ses clauses soient satisfaites.

Le **problème de satisfiabilité** est défini comme suit: étant donné une expression booléenne F, il faut déterminer s'il existe une affectation de valeurs 0 ou 1 aux $x_i, i = 1, ..., n$ telle que $F = 0$. S'il existe une telle affectation, alors on dit que le problème est **satisfiable**; sinon il est non satisfiable.

Le problème de satisfiabilité est important dans la théorie de la complexité et l'intelligence artificielle (Stan [25]).

Si l'on fait l'expansion des clauses en éliminant les compléments des variables,

alors on trouve une expression de la forme

$$C + f(X)$$

où C est une *constante* ≥ 0, $C + f(X) \geq 0 \ \forall X \in B^n$ et $f(X)$ est de la forme de la fonction-objectif du problème (P).

Un algorithme pour la résolution de (P) pourrait nous dire si $\max -f(X) = C$ (F satisfiable) ou $\max -f(X) < C$ (F non satisfiable).

Il faut remarquer tout d'abord que le problème (P) est NP-complet car il contient le problème quadratique ($\Omega = \{S, |S| = 1 \text{ou} 2\}$) qui est équivalent au problème de coupe de capacité maximum (Hammer [8]) qui est lui-même NP-complet (Garey, Johnson et Stockmeyer [5]).

Des algorithmes pour résoudre le problème (P) sont proposés depuis 1963. Hammer, Rosenberg et Rudeanu [11] proposèrent une méthode algébrique qui éliminait de façon récursive une variable à la fois; de telle sorte qu'elle produisait une fonction à chaque itération qui avait la même valeur optimale que la fonction à l'itération précédente. Crama, Hansen et Jaumard [4], en 1990, proposaient une amélioration de l'algorithme de [11] qui élimine la variable par un procédé de séparation et évaluation progressives. Au cours des années, d'autres méthodes, principalement énumératives ont été proposées. Citons entre autres Lawler et Bell [17], Mao et Wallingford [18] (méthodes d'énumération lexicographique), Berman [2], Hammer [9], Hammer et Peled [10] et Hansen [15] (méthodes de séparation et évaluation progressives).

C'est surtout le problème quadratique qui a reçu le plus d'attention surtout à cause de son importance pratique et de sa structure spéciale.

En plus des algorithmes de type séparation et évaluation progressives, de nombreuses approches utilisant des techniques continues (programmation linéaire ou dérivées partielles) ont été proposées (voir à cet égard le survol de Hansen [14] et l'article de Hammer et Simeone [13]).

Enfin, notons que Rosenberg [23] a démontré que le problème (P) est équivalent à un problème quadratique non contraint en variables $0-1$. Cette équivalence a un intérêt plutôt théorique puisque la transformation augmente de beaucoup le nombre de variables et de termes.

1.3 Le cas particulier.

Avant d'aborder le cas général de (P), on va considérer un cas particulier du problème (P) qui peut se résoudre de façon polynômiale (Picard et Ratliff [22])

À partir du problème (P), on en extrait les termes linéaires de la fonction-objectif qui prend la forme suivante:

$$\max f(X) = \sum_{i=1}^{n} a_i x_i + \sum_{S \in \Omega'} a_S \prod_{i \in S} x_i \tag{1.4}$$

où

$$\Omega' = \{S \in \Omega, |S| \geq 2\} \tag{1.5}$$

Dans le cas particulier qu'on va considérer, on suppose que:

$$a_S \geq 0; \forall S \in \Omega'. \tag{1.6}$$

Sans perte de généralité, on peut supposer que $a_i \leq 0, \forall i$; si non dans toute solution optimale $x_i = 1$. Ainsi l'on pourrait éliminer x_i du problème.

On définit maintenant un problème équivalent à (1.4) à (1.6), où on linéarise la fonction-objectif:

$$(\text{P1}) : \max f(X,Y) = \sum_{i=1}^{n} a_i x_i + \sum_{S \in \Omega'} a_S y_S \tag{1.7}$$

s.c :

$$y_S \leq x_i, \forall i \in S, \forall S \in \Omega' \tag{1.8}$$

$$y_S \in \{0,1\}, \forall S \in \Omega' \tag{1.9}$$

$$x_i \in \{0,1\}, i = 1,2,...,n. \tag{1.10}$$

Les contraintes (1.8), (1.9) et (1.10) forcent les deux conditions suivantes:

- Si $y_S = 1$, on aura automatiquement $x_i = 1; \forall i \in S$,
- Si $x_i = 1; \forall i \in S$, alors on aura $y_S = 1$ car $a_S \geq 0$, et l'on maximise.

Donc, $y_s = \prod_{i \in S} x_i = 1 \Leftrightarrow x_i = 1, \forall i \in S$ et $(P1)$ est équivalent à (P) pour ce cas particulier.

Dans le prochain chapitre nous définissons les outils qui nous permettront de résoudre ce cas particulier à l'aide des techniques de réseaux.

CHAPITRE 2

FERMETURE MAXIMALE DANS UN GRAPHE.

2.1 Rappels et Définitions

2.1.1 Fermeture maximale

On considère un graphe dirigé $G = (N, A)$, muni de pondérations c_i, pour tout nœud $i \in N$. On définit une **fermeture** $Y \subseteq N$ comme un sous-ensemble de nœuds du graphe tel que si un nœud i appartient à Y et qu'il existe un arc $(i, j) \in A$, alors le nœud j appartient aussi à Y.

La valeur de la fermeture Y est $\sum_{i \in Y} c_i$.

Le problème de la fermeture maximale de G consiste à trouver une fermeture de G dont la valeur est maximum. On illustre cette définition par un exemple.

2.1.2 Exemple 2.1

Dans le graphe de la Figure 2.1., on a par exemple les fermetures $\{1, 5, 9, 10, 7, 12, 11\}$ ou $\{2, 6, 5, 9, 10, 7, 11, 12, 3, 8, 4\}$ de valeurs 6 et 4 respectivement.

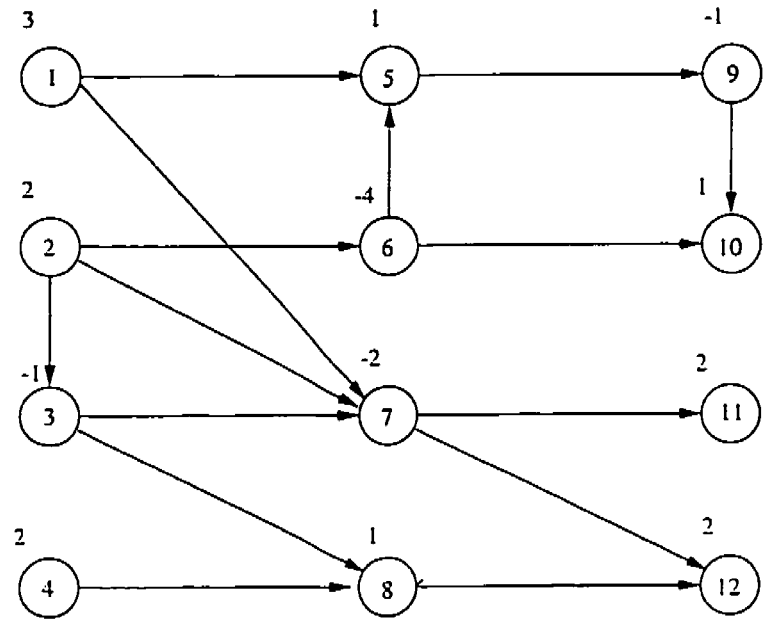


Figure 2.1 Exemple de graphe avec pondérations sur les nœuds

2.1.3 Formulation mathématique du problème de la fermeture maximale

Le problème de fermeture maximale peut se formuler comme suit:

$$(\text{PF}) \quad : \max \quad \sum_{i=1}^{n} c_i \, x_i \tag{2.1}$$

s.c:

$$x_i \leq x_j, \forall \, (i,j) \in A \tag{2.2}$$

$$x_i \in \{0,1\}, \; i = 1, \ldots, n \tag{2.3}$$

Il y a une variable x_i pour tout nœud $i \in N$ et une contrainte de précédence pour chaque arc $(i,j) \in A$). Les variables s'interprêtent comme $x_i = 1 \Leftrightarrow$ le nœud i est dans une fermeture.

Avant de présenter une méthode pour résoudre ce problème, nous rappelons le problème de flot maximum dans un réseau.

2.1.4 Problème de flot maximum dans un réseau

Soit un réseau $G = (N, A)$, où $|N| = n$ est le nombre fini de nœuds et $|A| = m$, le nombre d'arcs. Dans ce réseau on associe à chaque arc (i, j) une **borne inférieure** 0 et une **borne supérieure** u_{ij} sur la valeur du **flot** f_{ij}. Les u_{ij} sont des **nombres entiers** et le coût sur chaque arc est nul. Le problème de trouver un flot maximum entre deux nœuds s et t de G peut se formuler comme un problème de flot à coût minimum et peut être résolu par un algorithme de flot à coût minimum. Il existe plusieurs algorithmes d'étiquettage comme ceux de **Ford et Fulkerson** [6], **Karzanov** [24], ou **Dinic** [24] pour résoudre le problème de flot maximum et sont plus performants que l'algorithme du simplexe adapté aux réseaux ou d'autres algorithmes qui résolvent le problème de flot à coût minimum. Nous supposons une connaissance de l'algorithme de Ford et Fulkerson [6].

2.1.5 La méthode de Picard-Queyranne

Picard [20] résout le problème de fermeture maximale comme suit:

On crée un **réseau associé au graphe G** en ajoutant deux nœuds au graphe G dont le premier, la source, s, sera relié aux nœuds ayant une pondération positive tandis que les nœuds qui ont une pondération négative seront reliés au deuxième, le puits, t. Les arcs reliant s et t aux nœuds de G portent des capacités qui sont les valeurs absolues des pondérations dans G de ces nœuds. Les arcs intermédiaires, c'est-à-dire les arcs du graphe G, portent une capacité infinie.

Le réseau associé au graphe de l'exemple 2.1. est illustré à la figure 2.2. avec les capacités des arcs données à côté de ceux-ci.

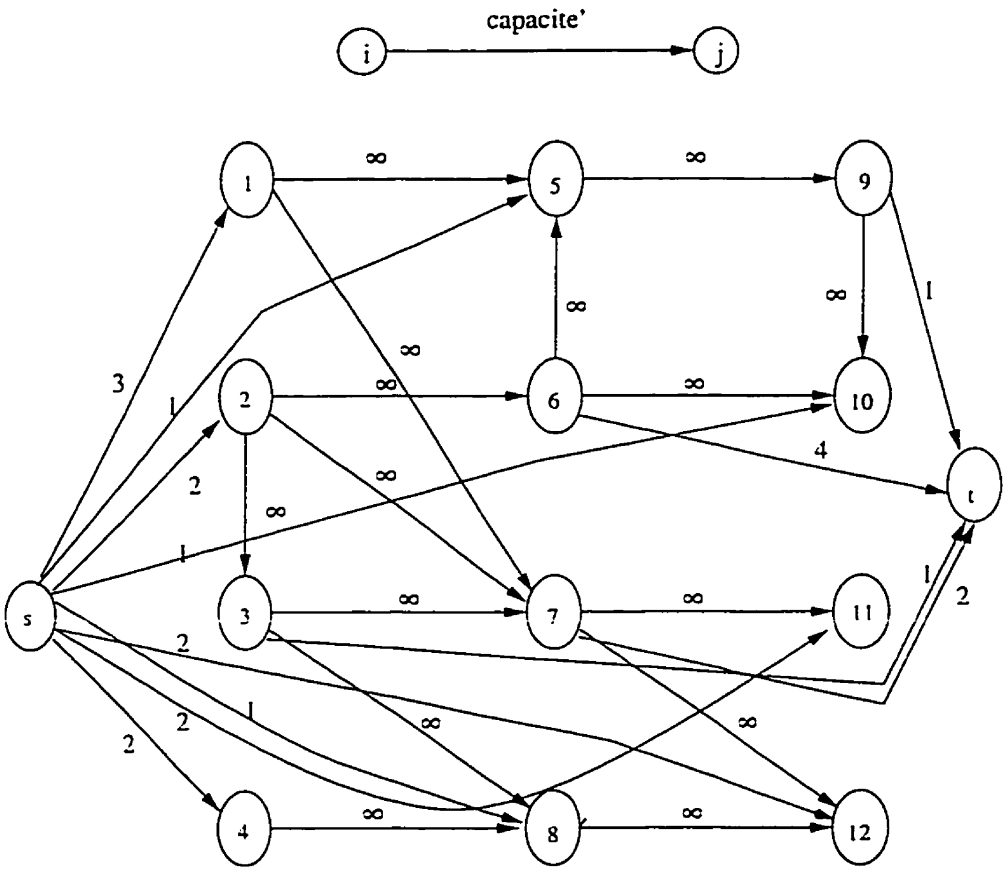


Figure 2.2 Réseau associé au graphe de la Figure 2.1.

On procède au calcul du flot maximum entre s et t dans ce réseau associé ainsi construit. Picard [20] démontre que la valeur de la fermeture maximale de G $= \sum_{c_i \geq 0} c_i -$ valeur du flot maximum. En d'autres termes, elle est égale à la différence entre la somme de toutes les capacités du côté de la source (capacité résiduelle à la source) et la valeur du flot maximum dans le réseau. Il démontre aussi que les nœuds de la fermeture maximale de G sont ceux (à part s) qui sont étiquetés à l'optimalité de l'algorithme de Ford et Fulkerson [6].

Le flot maximum pour l'exemple 2.1 est donné dans la Figure 2.3. ci-dessous où le flot non nul est indiqué à côté de l'arc qui le porte. Les nœuds étiquetés à la dernière itération de l'algorithme de Ford et Fulkerson [6] sont identifiés par un astérisque (*). Une fermeture maximale est alors $Y = \{1, 4, 5, 7, 8, 9, 10, 11, 12\}$ de valeur 9.

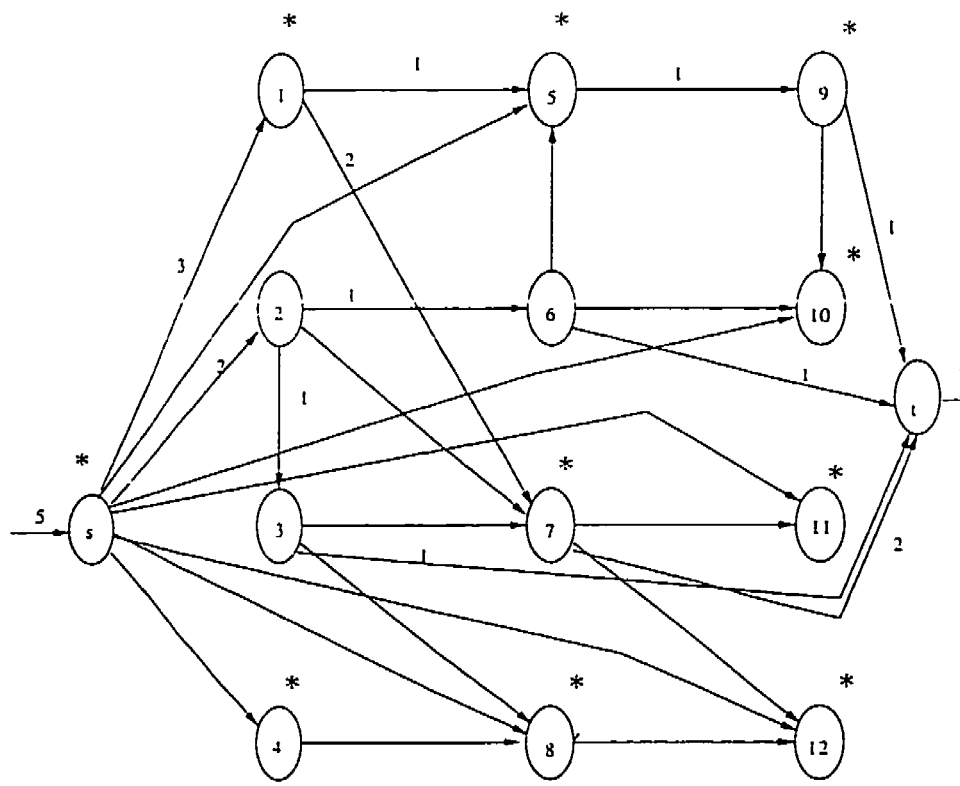


Figure 2.3 Flot maximum dans le graphe de la Figure 2.2.

2.1.6 Solution du cas particulier

On revient à la formulation du cas particulier

$$(\text{P1}) : \max f(X,Y) = \sum_{i=1}^{n} a_i x_i + \sum_{S\in\Omega'} a_S y_S \tag{2.4}$$

s.c:

$$y_S \leq x_i, \forall i \in S, \forall S \in \Omega' \tag{2.5}$$

$$y_S \in \{0,1\}, \tag{2.6}$$

$$x_i \in \{0,1\}, i = 1,2,...,n. \tag{2.7}$$

Il est évident que ce problème est équivalent à un problème (PF) de fermeture maximale dans le graphe $G = (N, A)$ où $N = \{1, 2, ..., n, S \in \Omega'\}$ et l'ensemble $A = \{(S, i), \forall S \in \Omega', \forall i \in S\}$. À l'optimalité du flot maximum, les nœuds i étiquetés correspondent aux variables x_i de $P1$ égales à 1. On peut noter que pour ce cas particulier, G est biparti. Par conséquent tout problème du type $(P1)$ peut être résolu par la méthode de Picard [20], et, puisque le problème de flot maximum est polynômial (Karzanov [24]), le cas particulier est polynômial.

On considère en guise d'exemple le problème suivant:

2.1.7 Exemple 2.2

$$\begin{aligned}\max f(X) = {} & -2x_1 - x_2 - 5x_3 - 2x_4 + 2x_1x_2 + 2x_1x_2x_3 \\ & +6x_1x_2x_4 + x_2x_3x_4 + x_1x_2x_3x_4\end{aligned} \tag{2.8}$$

s.c:

$$x_j \in \{0,1\}, i = 1,2,...,4. \tag{2.9}$$

et sa relaxation linéaire.

$$\begin{aligned} \max f(X,Y) &= -2x_1 - x_2 - 5x_3 - 2x_4 + 2y_{12} + 2y_{123} \\ &\quad +6y_{124} + y_{234} + y_{1234} \end{aligned} \tag{2.10}$$

$s.c:$

$$\begin{array}{lll} y_{12} \leq x_1 & y_{123} \leq x_1 & y_{124} \leq x_1 \\ y_{12} \leq x_2 & y_{123} \leq x_2 & y_{124} \leq x_2 \\ y_{234} \leq x_3 & y_{123} \leq x_3 & y_{124} \leq x_4 \\ y_{234} \leq x_4 & y_{1234} \leq x_1 & y_{1234} \leq x_2 \\ y_{234} \leq x_2 & y_{1234} \leq x_3 & y_{1234} \leq x_4 \end{array}$$

$$x_j \in \{0,1\}, i = 1,2,...,4.$$

$$y_s \in \{0,1\}.$$

Le graphe G et le réseau associé correspondants sont donnés aux Fig. 2.4. et 2.5.

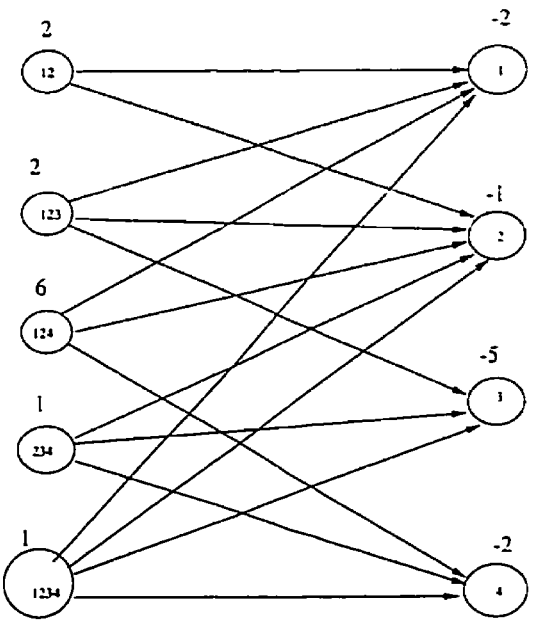


Figure 2.4 Graphe de l'exemple 2.2

Le flot maximal et les sommets étiquetés sont donnés à la Fig. 2.6.

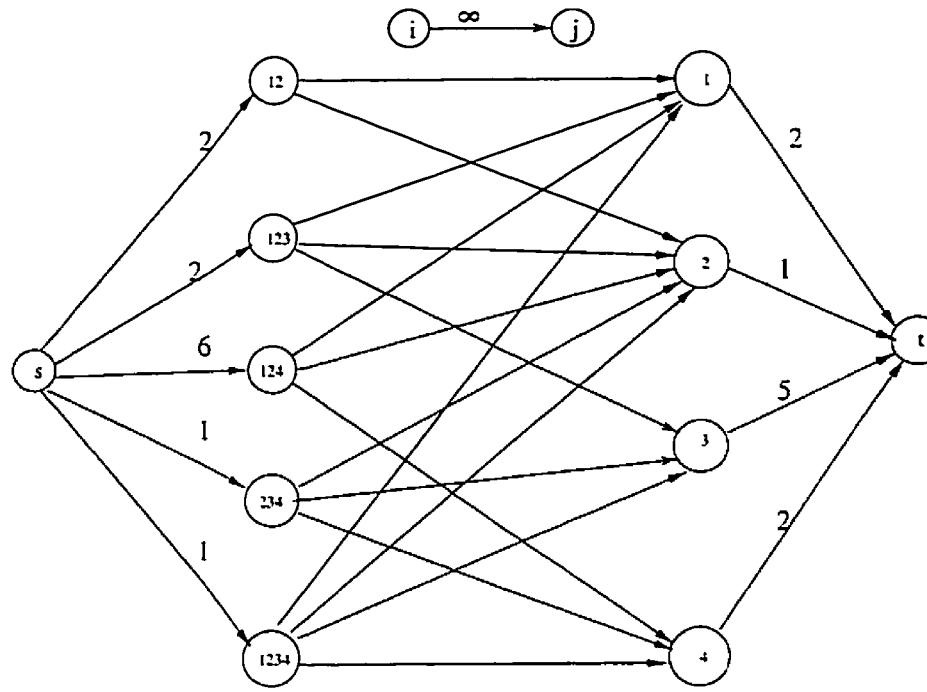


Figure 2.5 Le réseau associé au graphe de la Fig. 2.2.

On remarque sur la Fig. 2.6. l'ensemble de nœuds qui sont étiquetés, c'est-à-dire {12,124,1,2,4}. Ils constituent une fermeture maximale du graphe G considéré.

Donc, la valeur maximale de la fonction $f(X)$ de l'exemple 2.2 est 3 et elle est obtenue avec la solution $x_1^* = x_2^* = x_4^* = 1$ et $x_3^* = 0$.

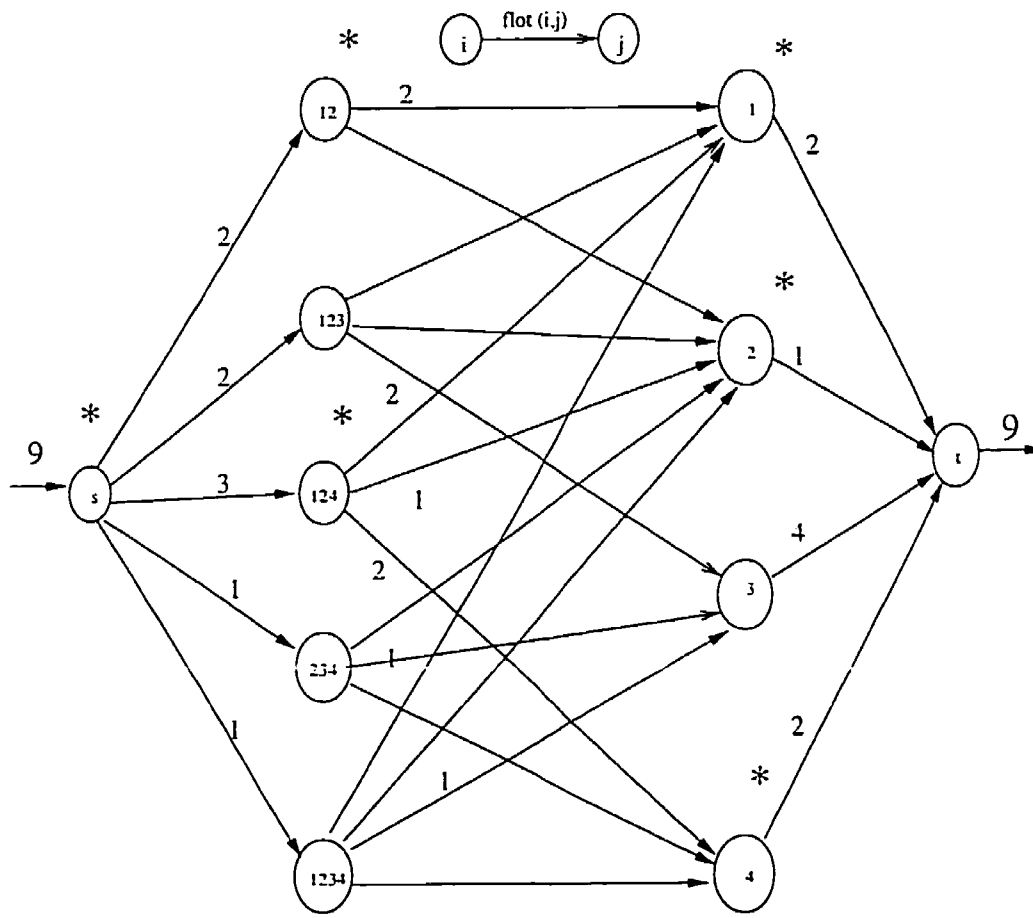


Figure 2.6 Les résultats du flot maximum pour l'exemple de la Fig. 2.4

CHAPITRE 3
PROBLÈME GÉNÉRAL

3.1 Relaxation du problème

On considère le problème général énoncé au chapitre 1.

$$(\mathrm{P}): \quad \max f(X) = \sum_{S \in \Omega} a_S \prod_{i \in S} x_i \tag{3.1}$$

tel que

$$x_i \in \{0,1\},\ i = 1, \ldots, n \tag{3.2}$$

où a_S est de signe quelconque. En mettant en évidence les termes linéaires et en effectuant une relaxation dans la fonction - objectif de manière à transformer les termes non linéaires en des termes linéaires, on définit le problème:

$$(P2) \quad \max f(X,Y) = \sum_{i=1}^{n} a_i x_i + \sum_{S \in \Omega'} a_S y_S \tag{3.3}$$

s.c :

$$y_S \leq y_{S'},\ \forall S,\ S' \in \Omega',\ S' \subseteq S \tag{3.4}$$

$$y_S \leq x_i,\ \forall\, i \in S,\ \forall\, S \in \Omega' \tag{3.5}$$

$$y_S \in \{0,1\},\ \forall S \in \Omega' \tag{3.6}$$

$$x_i \in \{0,1\},\ i = 1, 2, ..., n. \tag{3.7}$$

· On peut remarquer que les problèmes $(P2)$ et (P) ne sont pas équivalents. En effet, (X^*, Y^*), la solution optimale de $(P2)$ pourrait être irréalisable pour (P); en effet si $a_S < 0$, y_S^* peut être égal à 0 alors que $x_i^* = 1\ \forall i \in S$. Mais, $(P2)$ admet toutes les solutions de (P) et donc (X^*, Y^*) de $(P2)$ fournit une borne supérieure $f(X^*, Y^*)$ sur la valeur de $\max f(X)$ de (P), c'est-à-dire la relation suivante entre

les deux valeurs sera toujours vérifiée :

$$\max f(X) \leq \max f(X, Y).$$

On remarque que dans le cas particulier où $a_s \geq 0$, $\forall S \in \Omega'$, les contraintes (3.4) sont satisfaites pour n'importe quelle solution satisfaisant les contraintes (3.5), (3.6) et (3.7) ; car si $S' \subset S$ et si $y_S = 1$, alors d'après (3.5) $x_i = 1\ \forall i \in S$, ce qui implique que $x_i = 1\ \forall i \in S'$ et à l'optimalité de $(P2)$, $y_{S'} = 1$ car $a_{S'} \geq 0$. Donc on peut laisser tomber les contraintes (3.4) et l'on retrouve le problème (P1) du chapitre 1. Cette relaxation, $(P2)$, est due à Picard et Queyranne [21].

3.2 Description de l'algorithme

Dans la description de l'algorithme, nous supposons que les grandes lignes de la technique de séparation et évaluation progressives (SEP) sont connues. Dans cette section on précise la façon de calculer les bornes et la règle de branchement.

3.2.1 Calcul des bornes

Le problème (P2) est un problème de fermeture maximale (PF) pour un graphe G et donc peut être résolu par la méthode de Picard [20].

La valeur de $F(X, Y)$ va constituer la borne supérieure à chaque nœud de l'arbre de la SEP. Il y a cependant certaines définitions et simplifications qu'il est possible de faire auparavant.

La **forme standard** de $F(X)$ consiste à avoir tous les coefficients des termes linéaires négatifs ou nuls et les coefficients des termes non linéaires de signe quelconque. Pour y arriver, il faut éventuellement effectuer des complémentations des variables à coefficients positifs dans la partie linéaire en commençant par celle qui a le coefficient le plus grand par rapport aux autres (afin d'éviter trop de complémentations). Cela se fait en remplaçant la variable x_i par $1 - x_i$, et en

substituant cette expression dans tous les termes qui la contiennent. On réunit tous les termes semblables dans la fonction, puis on annule ceux qui ont les mêmes coefficients mais de signes opposés, enregistre la constante (le coefficient de la variable complémentée) ou on l'ajoute à celle qui existe déjà. Le processus est répété avec la nouvelle fonction et ce jusqu'à obtenir la forme standard. Ce processus doit se terminer car la constante croît à chaque complémentation et que la fonction-objectif est bornée. On garde cette forme de la fonction-objectif au cas où on tombe dans le cas particulier à un nœud de l'arbre de la SEP. À ce moment-là, comme nous l'avons vu, la borne supérieure correspond à la valeur de $F(X)$ à ce nœud.

Notons qu'après avoir mis la fonction originale $f(X)$ sous la forme standard, on obtient une nouvelle fonction, *constante* + $f'(X)$, où $f'(X)$, en forme standard, est une fonction différente de $f(X)$. Il faut donc tenir compte des complémentations nécessaires pour arriver à cette forme.

Or, tel qu'il est, le graphe associé à $(P2)$ n'est pas biparti mais peut le devenir moyennant certaines simplifications.
Par la suite, on dit qu'un nœud S' de G est **couvert** par un nœud S de G si l'arc $(y_S, y_{S'}) \in A$ (la contrainte $y_S \leq y_{S'}$ est dans $(P2)$, ou encore $S' \subset S$).
On met les nœuds $S \in \Omega'$ tels que $a_S > 0$ dans un ensemble N_1 et les nœuds $x_i, i = 1, 2, \cdots, n$ et $S \in \Omega'$ tels que $a_S < 0$ dans un ensemble N_2. Alors, on va montrer que dans le graphe $G = (N, A)$ de $(P2)$, on n'a qu'à considérer les arcs du graphe biparti $G' = (N, A')$ où $A' \subseteq N_1 \times N_2$.

Si $y_S, y_{S'} \in N_2$ et $(y_S, y_{S'}) \in A$ alors on peut éliminer $(y_S, y_{S'})$ de A car, à l'optimalité du flot maximum dans le réseau associé à G, s'il y a une **chaîne d'augmentation du flot** $s - y_{S_1} - y_S$, alors il y a une chaîne d'augmentation du flot $s - y_{S_1} - y_{S'}$ car S' est couvert par S. De plus S étant couvert par S_1, alors S' est couvert par S_1. Ainsi si y_S est étiqueté à l'optimalité du flot maximum dans le réseau associé à G, alors $y_{S'}$ l'est aussi (voir la Figure 3.1.).

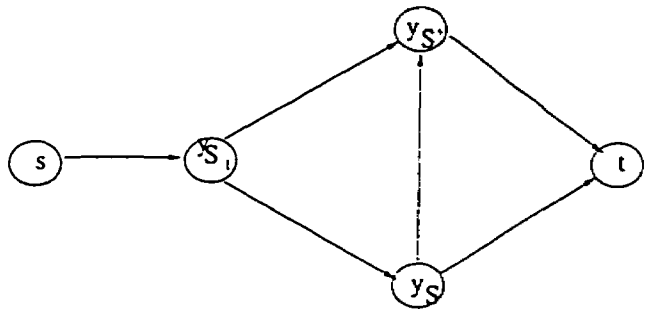


Figure 3.1 L'arc $(y_S, y_{S'})$ peut être éliminé.

Si $y_S, y_{S'} \in N_1$ et $(y_S, y_{S'}) \in A$, alors on peut éliminer $(y_S, y_{S'})$ de A. Tous les nœuds de N_2 qui sont couverts par S' le sont aussi par S. Donc à l'optimalité du flot maximum dans le réseau associé à G, si y_S est étiqueté, alors $y_{S'}$ l'est aussi, soit parce que $y_{S'}$ est étiqueté à partir de s, soit parce que l'arc $(s, y_{S'})$ est saturé et alors il existe du flot > 0 sur au moins un arc $(y_{S'}, y_{S''}), y_{S''} \in N_2$; donc $y_{S''}$ étant couvert par y_S peut être étiqueté et enfin $y_{S'}$ peut être étiqueté à partir de $y_{S''}$ (voir Figure 3.2.).

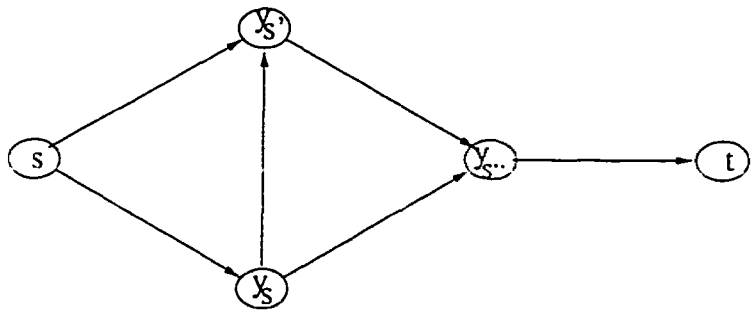


Figure 3.2 L'arc $(y_S, y_{S'})$ peut être éliminé.

Si $y_S \in N_2$ et $y_{S'} \in N_1$ et $(y_S, y_{S'}) \in A$, alors $(y_S, y_{S'})$ peut être éliminé de A. À l'optimalité de l'algorithme du flot maximum, si y_S est étiqueté, alors $y_{S'}$ l'est aussi; car il existe une chaîne d'augmentation de flot $s - y_{S_1} - y_S$ et si $y_{S'}$ n'est pas étiqueté de s, alors il existe un flot > 0 sur au moins un arc $(y_{S'}, y_{S''}) \in A$ et $y_{S''}$ est couvert par y_{S_1}, donc $y_{S'}$ est étiqueté par la chaîne d'augmentation de flot $s - y_{S_1} - y_{S''} - y_{S'}$ (voir Figure 3.3.).

Notons aussi que tout nœud $y_{S'} \in N_2$ qui n'est pas couvert par un nœud

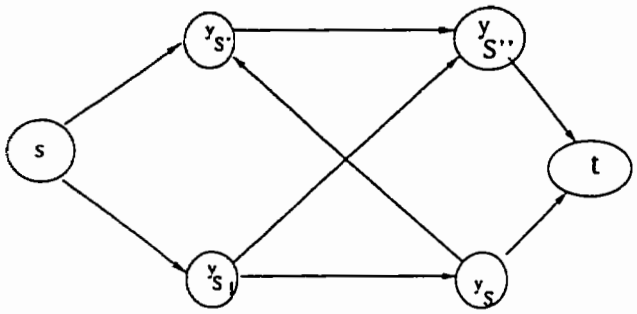


Figure 3.3 L'arc $(y_S, y_{S'})$ peut être éliminé.

$y_S \in N_1$ peut être éliminé de G car il n'y a aucune chaîne d'augmentation du flot possible de s à $y_{S'}$. Donc $y_{S'}$ ne sera jamais étiqueté à l'optimalité de l'algorithme de flot maximum, et par conséquent $y_{S'} = 0$ à ce nœud de l'arbre de la S.E.P..

On suppose que toutes les simplifications ci-haut détaillées ont été effectuées et que le graphe G associé à $(P2)$ est le graphe biparti G'.

3.2.2 Règle de branchement

On fait une **recherche en profondeur d'abord** le long des chemins $x_i = 1$. Pour choisir la variable de séparation, on regarde toutes les variables $x_i, i \in N' \subset N$ telles que $x_i = 1$ à l'optimalité du problème $(P2)$ du nœud actuel de l'arbre de la S.E.P., et l'on évalue la valeur de max $(constante + f(X, Y))$ avec $x_i = 0$.

Pour calculer ce maximum, il suffit de faire une **analyse de sensibilité** à partir du flot maximum dans le réseau. On donne à l'arc (x_i, t) une capacité infinie et l'on réoptimise. Ceci a l'effet de saturer l'arc incident à s dans toute chaîne d'augmentation de flot menant de s à x_i. Donc le nœud x_i ne pourra jamais être étiqueté à la réoptimalité; et alors $x_i = 0$ dans cette solution.

On fait cette analyse de sensibilité pour toutes les variables $x_i \in N'$, et l'on choisit comme variable de séparation celle qui fait décroître **le plus** la valeur de max $(constante + f(X, Y))$. L'idée, c'est que quand on remontera au nœud actuel de l'arbre et que l'on descendra la branche $x_i = 0$, on soit capable de sonder un nœud rapidement par le critère des bornes.

On tient compte de Z^*, la plus grande valeur de la constante dans la fonction-objectif $(constante + f(X, Y))$. Cette valeur constitue la meilleure borne inférieure trouvée à date. Notons que $X = 0, Y = 0$ correspond toujours à une solution de valeur 0 de la fonction $f(X, Y)$. On met à jour la valeur de Z^* quand on trouve une valeur plus grande de la constante.

Il y a une exception dans le choix de la variable de séparation. Si dans l'analyse de sensibilité, max $(constante + f(X, Y)) \leq Z^*$, pour un certain nombre de variables $x_i \in N'$, alors **toutes** ces variables peuvent être **fixées à la valeur** 1 à partir du nœud actuel de l'arbre de la S.E.P., parce que toute solution contenant ces variables égales à 0 est bornée par la valeur Z^*. À ce moment-là, on choisit comme variable

de séparation celle qui donne la plus petite valeur de $f(X,Y)$, telle que

$$\max\left(\textit{constante} + f(X,Y)\right) > Z^*.$$

Un nœud est **sondé** si:

1. $\max\left(\textit{constante} + f(X,Y)\right) \leq Z^*$
2. $f(X,Y)$ est telle que $a_S > 0,\ \forall y_S$ (le cas particulier).
3. $\max\left(\textit{constante} + f(X,Y)\right) = \textit{constante}$, c'est à dire $\max\ f(X,Y) = 0$.
   Les deux derniers cas correspondent à une solution optimale (X,Y) réalisable à (P).

3.3 Exemple

On considère la fonction non linéaire polynômiale suivante:

$$\begin{aligned}\max f(X) \;=\; & -3x_1 - 4x_2 - 5x_3 - 2x_4 - 6x_5 - 4x_6 + 6x_1x_2 - 2x_1x_3 + 4x_1x_4 \\ & +6x_1x_5 + 8x_2x_3 - 4x_2x_5 + 4x_2x_6 + 2x_3x_4 + 6x_3x_5 - 4x_4x_5 \\ & +2x_4x_6 + 8x_1x_3x_4 - 4x_2x_3x_4 - 4x_1x_5x_6 + 8x_2x_5x_6 - 4x_1x_2x_3x_4 \\ & -16x_1x_2x_3x_5 - 8x_1x_3x_4x_5 + 8x_1x_2x_5x_6 + 2x_3x_4x_5x_6 \\ & -5x_1x_2x_3x_4x_6 + 2x_1x_2x_3x_4x_5x_6; \qquad (3.8)\end{aligned}$$

s.c.

$$x_i \in \{0,1\},\ i = 1,2,...,6 \qquad (3.9)$$

On effectue la relaxation linéaire et l'élimination de certaines contraintes de précédence comme on l'a fait remarquer plus haut. Ceci nous donne deux ensembles disjoints de nœuds à partir desquels on construit un graphe biparti. On donne ci-après la forme simplifiée du problème.

$$\begin{aligned}\max f(X,Y) \;=\; & -3x_1 - 4x_2 - 5x_3 - 2x_4 - 6x_5 - 4x_6 + 6y_{12} - 2y_{13} + 4y_{14} \\ & +6y_{15} + 8y_{23} - 4y_{25} + 4y_{26} + 2y_{34} + 6y_{35} - 4y_{45} + 2y_{46} \\ & +8y_{134} - 4y_{234} - 4y_{156} + 8y_{256} - 4y_{1234} - 16y_{1235} - 8y_{1345} \\ & +8y_{1256} + 2y_{3456} - 5y_{12346} + 2y_{123456}; \end{aligned} \tag{3.10}$$

s.c.

$$\begin{array}{lllll} y_{12} \le x_1 & y_{26} \le x_2 & y_{15} \le x_1 & y_{23} \le x_2 & y_{34} \le x_3 \\ y_{12} \le x_2 & y_{26} \le x_6 & y_{15} \le x_5 & y_{23} \le x_3 & y_{34} \le x_4 \end{array}$$

$$\begin{array}{llll} y_{35} \le x_3 & y_{35} \le x_5 & y_{14} \le x_1 & y_{14} \le x_4 \\ y_{46} \le x_4 & y_{134} \le x_1 & y_{256} \le x_2 & y_{3456} \le x_3 \\ y_{46} \le x_6 & y_{134} \le x_3 & y_{256} \le x_5 & y_{3456} \le x_4 \\ y_{134} \le x_4 & y_{134} \le y_{13} & y_{256} \le x_6 & y_{3456} \le x_5 \end{array}$$

$$\begin{array}{llll} y_{1256} \le x_1 & y_{123456} \le x_1 & y_{256} \le y_{25} & y_{3456} \le x_6 \\ y_{1256} \le x_2 & y_{123456} \le x_2 & y_{3456} \le y_{45} & y_{123456} \le y_{156} \\ y_{1256} \le x_5 & y_{123456} \le x_3 & y_{123456} \le y_{13} & y_{123456} \le y_{1235} \\ y_{1256} \le x_6 & y_{123456} \le x_4 & y_{123456} \le y_{25} & y_{123456} \le y_{1345} \\ y_{1256} \le y_{25} & y_{123456} \le x_5 & y_{123456} \le y_{45} & y_{123456} \le y_{12346} \\ y_{1256} \le y_{156} & y_{1256} \le x_6 & y_{123456} \le y_{234} & y_{123456} \le y_{1234} \end{array}$$

$$y_S \in \{0,1\},$$

$$x_i \in \{0,1\},\ i = 1,2,...,6. \tag{3.11}$$

On crée le graphe biparti à partir de cette fonction et on calcule le flot maximum dans le réseau qui lui est associé.

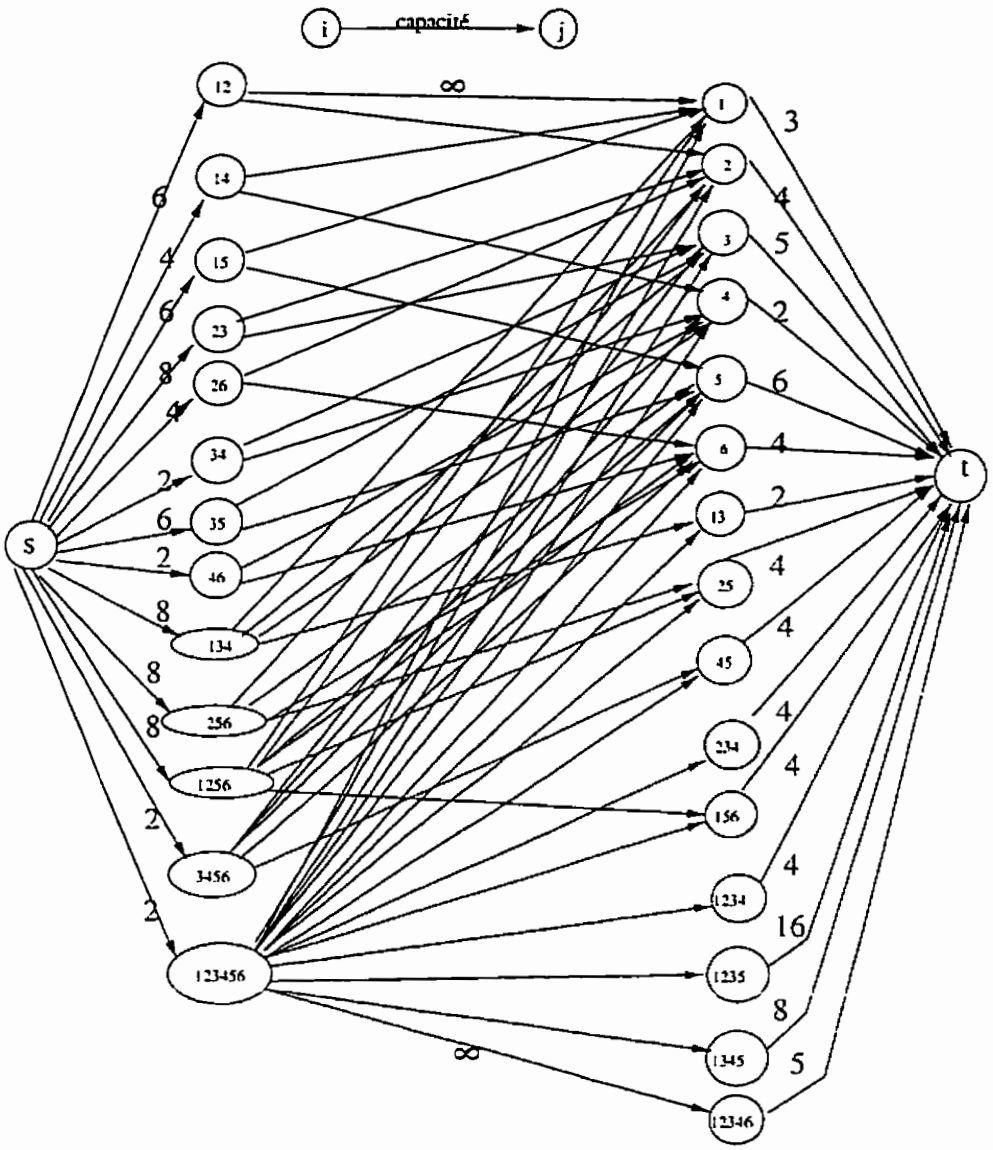


Figure 3.4 Le réseau associé au graphe construit à partir de f(X,Y).

Tous les arcs intermédiaires portent une capacité infinie.

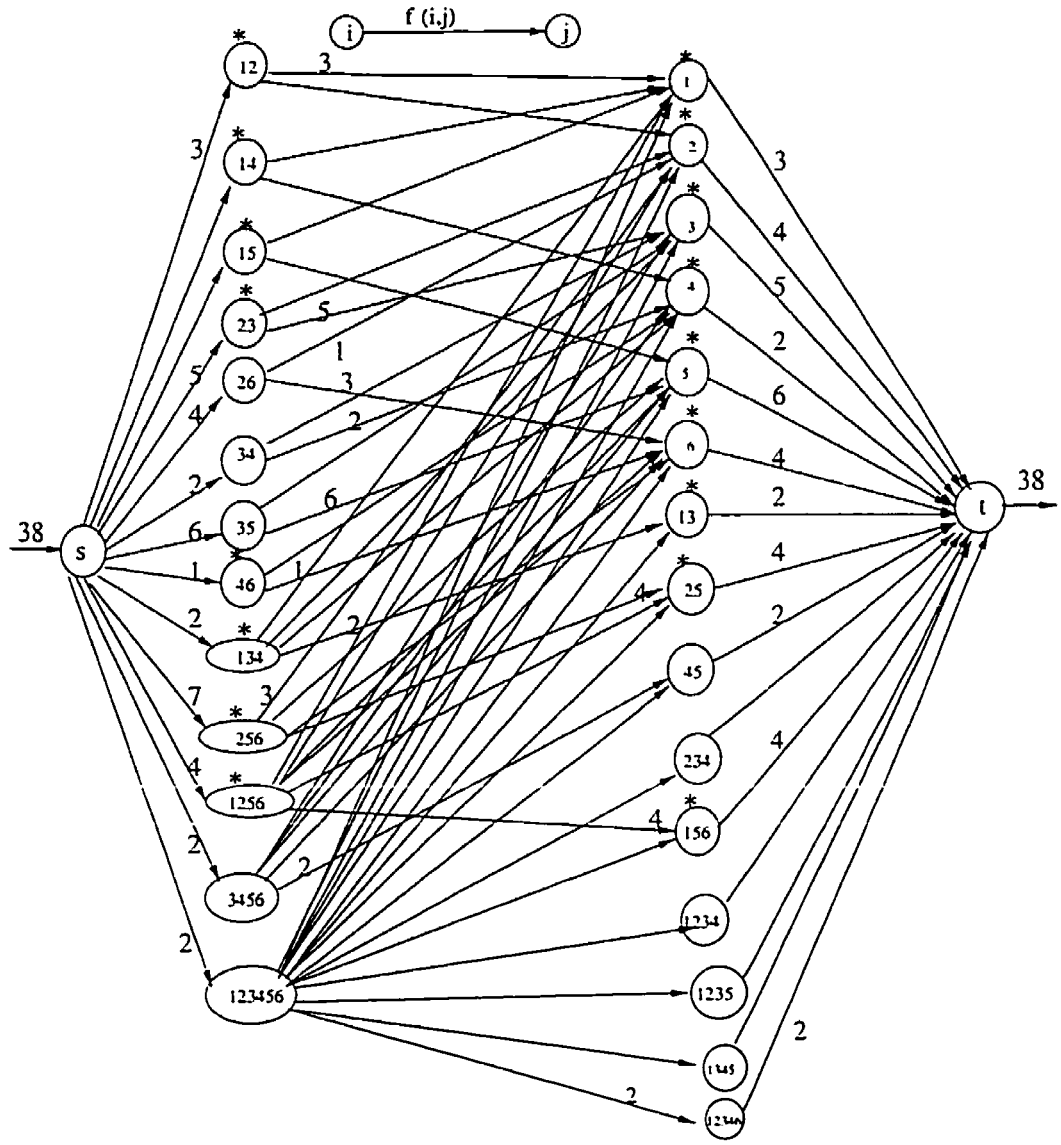


Figure 3.5 Les résultats du flot maximum avec le flot > 0 à côté de chaque arc.

La valeur de la fonction à ce niveau est égale à

$$\begin{aligned} \max f(X,Y) &= 0 + (66 - 38) \\ &= 28 \end{aligned}$$

et l'ensemble des variables qui valent 1 dans la fonction relaxée est:

$$\{x_1, x_2, x_3, x_4, x_5, x_6\}, \text{ et } \qquad Z^* = 0.$$

Nous allons maintenant effectuer une analyse de sensibilité sur les variables citées plus haut. On annule une à une ces variables précédemment choisies dans la première solution en augmentant la capacité à l'infini sur leurs arcs (x_i, t) correspondants dans le réseau. Les résultats obtenus sont enregistrés au fur et à mesure et sont comparés pour choisir la variable de séparation.

On trouve les résultats de l'analyse de sensibilité au tableau 3.1:

Tableau 3.1 Analyse de sensibilité sur les variables

$x_i = 0$	$\max f(X,Y)$
1	5
2	8
3	11
4	16
5	14
6	18

D'après ces résultats, on branche sur $x_1 = 1$.
On substitue $(x_1 = 1)$, dans la fonction $f(X,Y)$ (nécessitant la complémentation des variables x_2, x_4, x_3 , afin de regagner la forme standard) et la nouvelle fonction a la forme suivante:

$$\begin{aligned}\max f(X) \;=\; & 4 - 2x_2 - 3x_3 - 4x_4 - 26x_5 - 3x_6 - 4x_3x_5x_6 - 16x_2x_3x_5 \\ & -5x_2x_3x_6 - 2x_2x_3x_4x_5x_6 - 8x_3x_4x_5 - 5x_3x_4x_6 - 4x_4x_5x_6 \\ & -8x_2x_4 - 5x_2x_4x_6 - 18x_2x_5x_6 + 16x_5x_6 + 20x_2x_5 + x_2x_6 \\ & +12x_4x_5 + 2x_3x_4 + 4x_3x_4x_5x_6 + 8x_2x_3x_4 + 5x_2x_3x_4x_6 \\ & +2x_2x_3x_5x_6 + 5x_3x_6 + 3x_4x_6 + 18x_3x_5 + 2x_2x_4x_5x_6; \qquad (3.12)\end{aligned}$$

s.c.

$$x_i \in \{0,1\},\ i = 1,2,...,6. \qquad (3.13)$$

On calcule de nouveau le flot maximum dans le réseau associé au graphe construit à la base de cette fonction.

On trouve une nouvelle solution qui sera comparée à celle qui est déjà obtenue.

$$\begin{aligned}\max f(X,Y) \;&=\; 4 + 39 \\ &= 43 \qquad (3.14)\end{aligned}$$

avec

$$\begin{aligned}x_2^* &= 1. \\ x_3^* &= 1, \\ x_4^* &= 1, \\ x_5^* &= 1, \\ x_6^* &= 1; \qquad (3.15)\end{aligned}$$

Donc $Z^* = 4$ et les résultats sur l'analyse de sensibilité sont donnés dans le tableau 3.2.

Tableau 3.2 Analyse de sensibilité sur les variables

$x_i = 0$	$\max f(X,Y)$
2	20
3	17
4	26
5	0
6	17

La variable x_5 est nécessairement égale à 1 dans la fonction pour tous les nœuds successeurs du nœud actuel. On fixe la variable x_5 à 1 dans la fonction et complémente la variable qui a un coefficient positif en commençant par la variable qui a le plus grand coefficient; il s'agit seulement de la variable x_2 .
La nouvelle fonction est:

$$\begin{aligned}\max f(X) \;=\; & -4 - 18x_2 - 8x_2x_3x_4 - 3x_2x_3x_4x_6 - 4x_4x_6 - 4x_6 \\ & -2x_3x_6 - x_3 + 2x_3x_4 + 2x_3x_4x_6 + 16x_2x_3 + 8x_2x_4 + 17x_2x_6 \\ & +3x_2x_4x_6 + 3x_2x_3x_6 \qquad (3.16)\end{aligned}$$

On crée de nouveau un graphe à partir de la nouvelle fonction et on calcule le flot maximum qui passe à travers le réseau associé.
On obtient:

$$\max f(X^*, Y^*) \;=\; -4 + 22 = 18 \qquad (3.17)$$

avec

$$\begin{aligned} x_2^* &= 1 \\ x_3^* &= 1 \\ x_6^* &= 1; \qquad (3.18)\end{aligned}$$

Les résultats de l'analyse de sensibilité sont donnés dans le tableau 3.3.

Tableau 3.3 Analyse de sensibilité sur les variables

$x_i = 0$	$\max f(X,Y)$
2	3
3	1
6	7

Puisque $-4 + \max f(X,Y) < 4 = Z^*$ pour x_2, x_3 et x_6, on doit fixer ces variables à 1 dans la fonction à partir de laquelle on a calculé le flot maximum, et cette fonction devient:

$$\max f(X,Y) = 7, \text{ et } Z^* = 7.$$

Pour la branche $x_1 = 0$, on a déjà calculé $\max f(X,Y) = 5 < Z^* = 7$, donc ce nœud est sondé et la solution optimale a la valeur $Z^* = 7$.

Pour établir la solution optimale du problème, on remonte à la racine de l'arbre en retraçant les complémentations qui ont eu lieu sur les variables.

La solution optimale de cet exemple est donnée par les variables:

$$x_1^* = 1, x_2^* = 1, x_3^* = 0, x_4^* = 1, x_5^* = 1, x_6^* = 1$$

et

$$\max f(X^*) = 7$$

On résume l'arbre de la S.E.P. pour cet exemple dans la figure 3.6.

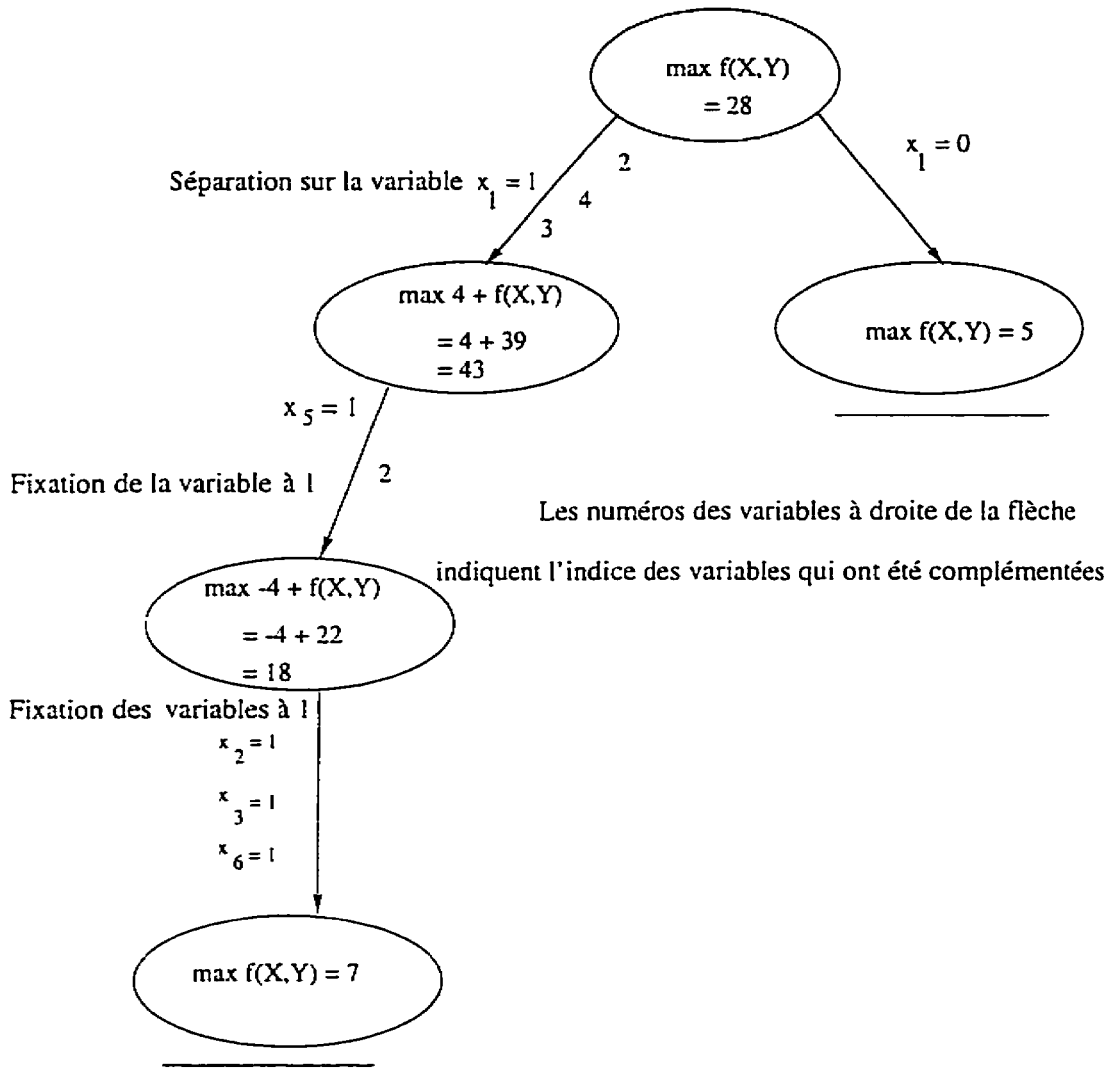


Figure 3.6 L'arbre de branchement pour l'exemple.

CHAPITRE 4
LES RÉSULTATS NUMÉRIQUES

Nous présentons dans ce chapitre les résultats obtenus sur les tests réalisés sur des problèmes créés de manière aléatoire ou ceux qu'on a pu trouver dans certains articles et manuels qui parlent de la programmation non-linéaire en variables $0-1$. Nous avons construit des fonctions qu'on maximisait d'une part ou qu'on minimisait d'autre part sachant que

$$\min f(X) = -\max -f(X)$$

$$X \in B^n.$$

Nous avons élaboré un programme en language C qui crée le graphe à partir de la fonction relaxée. La programmation a été faite sur une machine *Sun* **SPARCstation** du Laboratoire du **CRT** (Centre des Recherches sur les Transports , Université de Montréal). Nous donnons ci-après les détails de ce qui a été réalisé au niveau de la programmation en vue de tester la performance de l'algorithme de Picard-Queyranne [21].

Dans ce programme, nous avons utilisé les structures de listes chaînées et de pointeurs. Cela nous a facilité à construire un tableau représentant la fonction donnée. Nous enregistrons la valeur de l'indice, représentant la variable la plus grande dans la fonction, puis la liste des termes à coefficient négatif et celle des termes à coefficient positif. La **fonction** ainsi construite est composée par une liste de termes et chaque **terme** est précédé par un **signe**, un **coefficient** en valeur absolue et la liste d' indices des variables du terme. On note que chaque terme va correspondre à un nœud du graphe associé à la fonction.

Comme on l'a vu dans la section 3.2.1, le graphe associé à la fonction-objectif peut être ramené à un graphe biparti. Le programme effectue cette réduction.

Nous inscrivons le graphe biparti dans un fichier de sortie, appelé "*output*". Le graphe construit sert de données (ou "*input*") à une sous-routine de flot maximum qui est un algorithme de **pré-flot** dû à Karzanov [24]. Ce code a été implémenté par M. Beyime Tachefine, étudiant au doctorat du département de mathématiques et de génie industriel. Le code s'avère être très efficace, cela dépend de la performance de la machine: la vitesse et la mémoire vive.

On note que dans ce programme, après avoir calculé le flot maximum, il fournit également la fermeture maximale dans le graphe et la liste des nœuds qui sont étiquetés à l'optimalité. À partir de ces nœuds étiquetés, ceux qui correspondent aux termes linéaires seront fixés un à un égal à 0, en augmentant la capacité à l'infini sur les arcs qui les relient au puits. Le programme calcule de nouveau le flot maximum à partir d'un flot nul, bien qu'il soit possible de regagner l'optimalité à partir du flot maximum précédent. On a fait ce choix pour éviter de faire des modifications au code de Tachefine. Ensuite notre programme présente les résultats de l'analyse de sensibilité dans un tableau des fermetures maximales pour les variables correspondantes aux termes linéaires dont les nœuds correspondants sont étiquetés. Il spécifie également la fermeture dont la valeur est la plus petite.

Pour le branchement, nous **fixons la variable de séparation à 1** dans la fonction. Nous **effectuons des complémentations** si c'est nécessaire, puis nous remettons la fonction sous la **forme standard** et nous **préparons le graphe de nouveau** si nécessaire (dans le cas d'analyse de sensibilité).

On a construit en outre une fonction dans le même programme qui évalue la fonction lorsque la variable de séparation $x_i = 0$ et qui donne une nouvelle fonction sous la forme standard. La différence est qu'on détruit tous les termes qui contiennent la variable x_i et on imprime la fonction. Cette nouvelle fonction sera traitée de la même manière que la branche $x_i = 1$.

Les seules interventions humaines dans l'application de la technique de séparation et évaluation progressives sont:

1. Choix de la variable de branchement et fixation des variables à 1 ou 0.
2. Application des règles de sondage.
3. Mise-à-jour de la meilleure valeur de la fonction-objectif, Z^*.
4. Les déplacements dans l'arbre de la S.E.P.

En ce qui concerne les tests, nous avons engendré un certain nombre de fonctions. Nous avons généré des coefficients des termes dans l'intervalle [-99, 99], et les indices des termes dont le degré est fixe sont engendrés aléatoirement entre 1 et n. Le signe des termes linéaires est toujours "-", tandis que les termes non linéaires portent des signes "+" et "-" en alternance.

En ce qui concerne les exemples tirés des références, nous attirons une attention particulière au lecteur qui s'intéresse à ces exemples que nous avons vérifiés avec notre programme afin qu'il puisse retrouver leurs données dans la référence correspondante écrite à côté de nos résultats. On peut alors trouver ces exemples dans les références à la page 37.

On présente ci-après le tableau représentant les résultats numériques à la page suivante. Dans ce tableau, on a utilisé le paramètre l que nous définissons comme suit:

$$l = \sum_{k=2}^{n} \frac{(\text{nombre de termes de degré } k) \times k}{\text{nombre total de termes non linéaires}}$$

l est le degré moyen non linéaire de la fonction-objectif

Tableau 4.1 Résultats numériques des fonctions de 4 à 8 variables

Nombre de variables (n)	Nombre de termes dans la fonction $f(X)$	degré maximum du terme dans la fonction $f(X)$	Nombre moyen de variables dans un terme non linéaire (l)	Nombre de sommets dans l'arbre	
4	12	4	2.7	2	
	10	4	2.6	2	[20]
	9	3	3	2	[20]
	8	4	2.7	2	
	8	3	2.4	2	[26]
5	24	5	4	2	
	21	5	3.2	3	
	19	4	2.7	3	
	19	4	3.2	3	
	15	5	2.9	2	
	13	3	2.1	2	[7]
	13	2	2	1	[7]
	9	2	2	1	
	9	2	2	2	
	7	3	2.2	2	[12]
	7	3	2.8	2	[12]
6	28	6	3	4	
	26	5	3.1	4	
	24	6	2.8	2	[12]
	20	6	3.4	3	
	15	2	2.4	2	[4]
	11	3	2.4	1	[4]
7	31	5	2.8	2	[12]
	25	5	2.8	2	
8	10	3	2.3	3	[12]

Tableau 4.2 Résultats numériques sur les fonctions de 10 à 100 variables

Nombre de variables (n)	Nombre de termes dans la fonction $f(X)$	degré maximum du terme dans la fonction $f(X)$	Nombre moyen de variables dans un terme non linéaire (l)	Nombre de sommets dans l'arbre
10	55	2	2	2
	51	5	2.5	4
12	120	15	10	32
	51	5	2.6	3
	50	5	2.6	5
	47	5	2.5	4
15	60	7	2.8	6
20	300	2	2	16
	250	2	2	9
	200	15	10	128
	200	2	2	7
	160	2	2	43
	150	2	2	7
	120	12	8	32
	112	3	3	4
	109	3	2.3	20
	100	2	2	5
	50	2	2	2
30	500	2	2	81
	300	15	10	152
	200	2	2	10
	100	2	2	6
40	200	6	5	134
50	300	12	8	186
	200	2	2	33
	150	15	8	111
100	250	3	2.3	10
	200	2	2	6

CONCLUSION

Dans ce mémoire, nous avons vérifié la performance de l'algorithme, la technique de séparation et évaluation progressives pour optimiser une fonction non linéaire en variables 0-1. Nous avons remarqué que pour les fonctions de petite taille, l'algorithme converge rapidement vers la solution optimale. Mais pour les fonctions de grande taille, le nombre de sommets de calcul semble augmenter considérablement. Le nombre de ces sommets d'évaluation de la fonction augmente, c'est tout à fait normal. On note en passant que pour une fonction f(X), le nombre de solutions possibles est de l'ordre de 2^n, et aussi longtemps que n augmente, le nombre de solutions possibles augmente aussi. Nous sommes confiants à de très bons résultats du moment où l'algorithme fonctionne sans qu'il y ait intervention humaine, et à ce-moment-là, le temps de calcul deviendrait le critère de comparaison.

Un test que nous n'avions pas le temps de faire, serait de brancher sur $x_i = 0$ pour essayer d'avoir rapidement une bonne solution $Z^* > 0$ qui permettrait de sonder des nœuds des branches $x_i = 1$.

Bien que les cas quadratiques du problème (P) revêtent un intérêt particulier à cause de leurs nombreuses applications, nous n'avons pas testé ces cas exclusivement, notre but était de constater la performance de l'algorithme sur les cas généraux.

Nous notons, cependant, que dans presque tous les cas considérés, la fonction était "presque quadratique" ($l \approx 2$), et dans ces cas même pour $n = 20$, la convergence était rapide.

Il y a peut être d'autres critères de branchement et d'accélération de sondage qu'on pourrait prouver applicables aux cas quadratiques, mais ceux-ci doivent attendre des travaux futurs.

RÉFÉRENCES

[1] BARAHONA, F. (1986). A Solvable Case of Quadratic 0-1 Programming, *Discrete Applied Mathematics* **13**, 23-26.

[2] BERMAN, G. (1969). A Branch and Bound Method for Maximization of Pseudo-Boolean Functions, *Faculty of Mathematics, University of Waterloo*, Canada.

[3] BLAIR, C.E., JEROSLOW, R. G. et LOWE, J.K. (1986). Some Results and Experiments in Programming Techniques for Propositional Logic, *Comput. & Ops. Res.* **13**, No. 5, 633-645.

[4] CRAMA, Y., HANSEN, P. et JAUMARD, B. (1990). The Basic Algorithm for Pseudo-Boolean Programming Revisited, *Discrete Applied Mathematics* **29**, 171-185.

[5] GAREY, M.R., JOHNSON, D.S. et STOCKMEYER, L. (1976). Some Simplified NP-Complete Graph Problems, *Theor. Comp. Sci.* 1, 231-267.

[6] FORD, L.R. et FULKERSON, D.R. (1962). Flows in Networks, *Princeton University Press.*

[7] GULATI, P., GUPTA, S.K. et MITTAL, A.K. (1984). Unconstrained Quadratic Bivalent Programming Problem, *European Journal of Operational Research* **15**, 121-125.

[8] HAMMER, P.L. (1965). Some Network Flow Problems Solved with Pseudo-Boolean Programming, *Operations Research* **13**, 388-399.

[9] HAMMER, P.L. (1971). A B.B.B. Method for Linear and Nonlinear Bivalent Programming, in : B. Avi-Itzhak, *Developments in Operations Research, Gordon and Breach,* New York, 45-82.

[10] HAMMER, P.L.et PELED, U. (1972). On the Maximization of Pseudo-Boolean Functions, *JCM* **19**, 265-282.

[11] HAMMER, P. L., ROSENBERG, I. et RUDEANU, S. (1963). On the Determination of the Minima of Pseudo-Boolean Functions, *Stud. Cer. Math.* **14**, 359-364.

[12] HAMMER, P.L. et RUDEANU, S. (1968). Boolean Methods in Operations Research and Related Areas, Heidelberg, *Springer Verlay.*

[13] HAMMER, P.L. et SIMEONE, B. (1987). Quadratic Functions of Binary Variables, *RUTCOR Research Report No 20-87,* Rutgers University.

[14] HANSEN, P. (1979). Methods of Nonlinear 0-1 Programming, *Annals of Discrete Mathematics* 5, 53-70.

[15] HANSEN, P. (1969). Un Algorithme S.E.P. pour les Programmes Pseudo-Booléens Non linéaires , *Cahiers, Centre Études Recherche Opérationnelle* 11, 26-44.

[16] KALANTARI, B., BAGCHI, A. (1990). An Algorithm for Quadratic Zero-One Programs, *Naval Research Logistics* **37**, 527-538.

[17] LAWLER, E. L. et BELL, M.D. (1966). A Method for Solving Discrete Optimization Problems, *Operations Research* **14**, 1098-1112.

[18] MAO, J.C.T. et WALLINGFORD, B.A. (1968). An Extension of Lawler and Bell's Method of Discrete Optimization with Examples from Capital Budgeting, *Management Sci.* **15**, 51-60.

[19] PAUL A. JENSEN and J. Wesley BARNES, (1980). Network Flow Programming, *Operations Research Group.*

[20] PICARD, J.C. (1976). Maximal Closure of a Graph and Applications to Combinatorial Problems, *Management Sci.* **22**, 1268-1272.

[21] PICARD, J.C. et QUEYRANNE, M. (1982). A Network Flow Solution to some Nonlinear 0-1 Programming Problems, *Networks* **12**, 141-159.

[22] PICARD, J.C., et RATLIFF, H.D. (1975). Minimum Cuts and Related Problems, *Networks* **5**, 357-370.

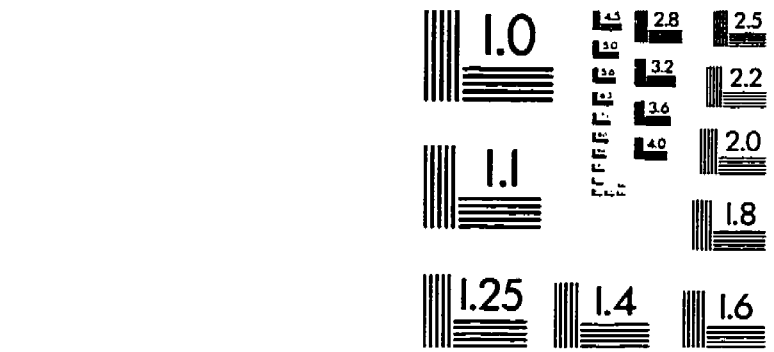
[23] ROSENBERG, I. (1975). Reduction of Bivalent Maximization to the Quadratic Case, *Cahiers Centre Études Recherche Opérationnelle* **17**.

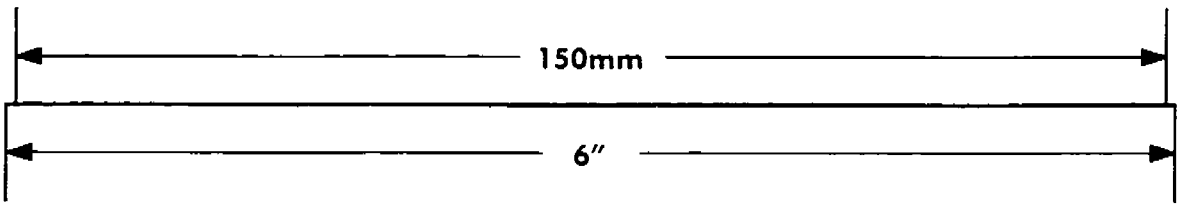
[24] RAVINDRA, K. A., MAGNANTI, T. L., et ORLIN, J. B. (1993). Network Flows: Theory, Algorithms, and Applications, *Printice-Hall, Inc.*, Englewood Cliffs, New Jersey 07632.

[25] STAN, M. (1996). Tournées de Véhicules et Satisfaisabilité Logique, *Thèse de Doctorat, École Polytechnique de Montréal.*

[26] WARREN, E.A., BILLIONNET A. and SUTTER, A. (1990). Unconstrained 0-1 Optimization and Lagrangean Relaxation, *Discrete Applied Mathematics* **29**, 131-142, North-Holland.

TEST TARGET (QA-3)





APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989