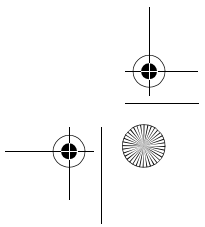
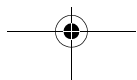
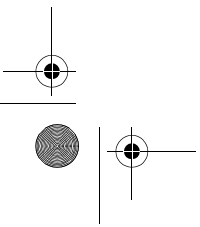
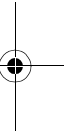
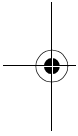
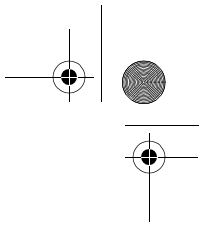
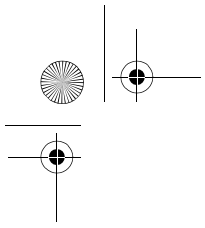




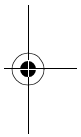
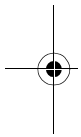
Table des matières

Avant-propos	XIII
Introduction	1
1. La programmation linéaire en variables mixtes.....	1
2. La programmation quadratique convexe en variables mixtes	3
3. Formulation d'un problème d'optimisation par un programme mathématique.....	4
4. Transformation d'un programme mathématique en un programme linéaire ou quadratique convexe en variables mixtes	6
5. Complexité des algorithmes et des problèmes	9
6. Les solveurs et les langages de modélisation	12
7. Les différentes étapes de la résolution d'un problème d'optimisation	19
8. Exemples de formulations	21
9. Résolution approchée d'un programme mathématique non linéaire par un programme linéaire en variables mixtes	38
Chapitre 1 – Programmation linéaire et programmation quadratique convexe	41
1. Introduction.....	41
2. Exemples d'application de la programmation linéaire et de la programmation quadratique convexe, interprétation géométrique	44
2.1. <i>Un premier exemple de formulation par la programmation linéaire :</i> <i>la détermination d'un plan optimal de fabrication</i>	44
2.2. <i>Un deuxième exemple de formulation par la programmation linéaire :</i> <i>l'optimisation du rapport de deux fonctions linéaires</i>	46



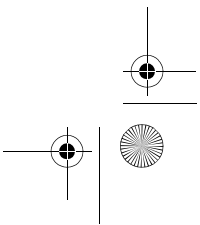
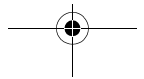
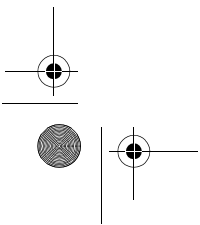


Introduction



Quelle est la meilleure solution d'un problème ? Cette question a été étudiée dans de nombreux contextes. Quelle est, par exemple, la courbe du plan la plus courte qui rejoint deux points donnés de ce plan ? Quelle est la courbe la plus courte rejoignant deux points donnés à la surface d'une sphère ? Quelle est le nombre minimal de couleurs nécessaires pour colorier les départements d'une carte de France, de façon à ce que deux départements ayant une frontière commune n'aient pas la même couleur ? Pour un périmètre donné, quel est le rectangle de surface minimale ? Étant donné un ensemble de villes réparties géographiquement, quel est le meilleur parcours qui permet de les visiter toutes, une fois et une seule ? En quelle quantité et avec quelle fréquence faut-il commander un article de façon à satisfaire la demande de la clientèle et à minimiser les coûts de gestion du stock ? Des problèmes de ce type sont appelés problèmes d'optimisation. D'après Robert Faure, Jean-Paul Boss et André Le Garff, ce n'est pas d'aujourd'hui que l'homme cherche à optimiser les résultats qu'il peut obtenir dans des conditions déterminées. Ils en donnent pour preuve l'installation de Didon sur la côte africaine, contée par Virgile au livre I de l'Énéide : les habitants lui concédaient autant de terre qu'elle en pourrait enclore au moyen d'une lanière tirée de la peau d'un seul taureau. D'après ces auteurs, la reine connaissait en l'an 814 avant notre ère la figure plane qui, à périmètre donné, présente la surface maximale, puisque Carthage fut bâtie en arc de cercle autour de sa citadelle.

1. LA PROGRAMMATION LINÉAIRE EN VARIABLES MIXTES



Parmi tous les problèmes d'optimisation difficiles, il y en a un que l'on sait particulièrement bien traiter, c'est la maximisation (ou la minimisation) d'une fonction linéaire $f(x_1, x_2, \dots, x_n)$, les variables x_1, x_2, \dots, x_n étant soumises à des contraintes

linéaires. Les variables sont des variables réelles, mais certaines d'entre elles peuvent être astreintes à ne prendre que des valeurs entières. De façon générale, ce problème d'optimisation peut s'écrire sous la forme P1, donnée ci-après, où R désigne l'ensemble des nombres réels, R^+ , l'ensemble des nombres réels positifs ou nuls et N , l'ensemble des entiers positifs ou nuls (ou naturels). On cherche à maximiser la fonction linéaire de n variables, $c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n$, les variables étant soumises aux m contraintes linéaires 1.1. Tous les coefficients du problème, c'est-à-dire les coefficients de la fonction à maximiser, $c_0, c_1, c_2, \dots, c_n$, les coefficients des membres de gauche des contraintes, $a_{11}, a_{12}, \dots, a_{1n}, a_{21}, a_{22}, \dots, a_{2n}, \dots, a_{m1}, a_{m2}, \dots, a_{mn}$, et les seconds membres de ces contraintes, b_1, b_2, \dots, b_m , sont des nombres réels. Il y a trois sortes de variables : les variables x_1, x_2, \dots, x_p qui peuvent prendre a priori n'importe quelle valeur réelle, les variables $x_{p+1}, x_{p+2}, \dots, x_q$ qui ne peuvent prendre que des valeurs réelles positives ou nulles et, enfin, les variables $x_{q+1}, x_{q+2}, \dots, x_n$ qui ne peuvent prendre que des valeurs entières positives ou nulles.

$$(P1) \left\{ \begin{array}{l} \max \quad f(x_1, x_2, \dots, x_n) = c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n \\ \text{s.c.} \quad \left\{ \begin{array}{ll} a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n \leq b_i & i = 1, \dots, m \quad (1.1) \\ x_j \in R & j = 1, \dots, p \quad (1.2) \\ x_j \in R^+ & j = p+1, \dots, q \quad (1.3) \\ x_j \in N & j = q+1, \dots, n \quad (1.4) \end{array} \right. \end{array} \right.$$

Le problème P1, que l'on appelle programme linéaire en variables mixtes, consiste donc à attribuer des valeurs réelles aux variables x_1, x_2, \dots, x_p , des valeurs réelles positives ou nulles aux variables $x_{p+1}, x_{p+2}, \dots, x_q$, et des valeurs entières positives ou nulles aux variables $x_{q+1}, x_{q+2}, \dots, x_n$, de façon à maximiser la fonction linéaire $c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n$, tout en respectant les m contraintes linéaires $a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n \leq b_i$ ($i = 1, \dots, m$). Le problème P2 donné ci-après est une instance de P1, c'est-à-dire un problème de type P1, comportant 5 variables et 4 contraintes, sans compter les contraintes portant sur la nature des variables.

$$(P2) \left\{ \begin{array}{l} \max \quad f(x_1, x_2, \dots, x_5) = 4 + 2x_1 + 20x_2 + 20x_3 + 7x_4 + 80x_5 \\ \text{s.c.} \quad \left\{ \begin{array}{ll} 10x_1 + 3x_2 + 8x_3 + 2x_4 + 2x_5 \leq -2000 & (2.1) \\ 5x_2 + 3x_3 + 2x_4 + 6x_5 \leq 30 & (2.2) \\ 5x_1 + 4x_2 - 2x_4 + 4x_5 \leq -7000 & (2.3) \\ -x_2 + 10x_3 + 9x_5 \leq 10 & (2.4) \\ x_1 \in R & (2.5) \\ x_2, x_3 \in R^+ & (2.6) \\ x_4, x_5 \in N & (2.7) \end{array} \right. \end{array} \right.$$

type de fonction économique (linéaire ou quadratique convexe). Elles indiquent également que les programmes considérés comportent à la fois des variables réelles et des variables entières. Lorsqu'on étudie un problème d'optimisation, deux cas sont envisageables : la formulation naturelle du problème est déjà un PLVM ou un PQCVM ou, au contraire, elle ne l'est pas. Nous nous intéresserons plus particulièrement dans cet ouvrage au second cas. Nous verrons en particulier comment transformer certains programmes mathématiques non linéaires en PLVM.

Réécrivons sous la forme P4 le problème général de programmation non linéaire en variables mixtes P3. Les fonctions g et h_i ($i = 1, \dots, m$) sont des fonctions quelconques, et $x = (x_1, \dots, x_n)$ est le vecteur des inconnues. Nous supposons que P4 admet une solution optimale. Nous nous intéressons dans cet ouvrage aux linéarisations de P4, c'est-à-dire à la formulation de P4 par un PLVM équivalent. De façon générale, notons $S_{\text{opt}}(\Pi)$ l'ensemble des solutions optimales d'un programme mathématique Π .

$$(P4) \begin{cases} \min g(x_1, \dots, x_n) \\ \text{s.c.} \begin{cases} h_i(x) \leq 0 & i = 1, \dots, m & (4.1) \\ x_j \geq 0 & j = 1, \dots, r & (4.2) \\ x_j \in \{0, 1\} & j = r + 1, \dots, s & (4.3) \\ x_j \in N & j = s + 1, \dots, n & (4.4) \end{cases} \end{cases}$$

Généralement, nous dirons que le programme linéaire en variables mixtes P5, portant sur les variables x_1, \dots, x_n et sur de nouvelles variables y_1, \dots, y_l , est une linéarisation du programme P4 si P5 admet une solution optimale et si la propriété suivante est vérifiée :

$$(x^*, y^*) \in S_{\text{opt}}(P5) \Rightarrow x^* \in S_{\text{opt}}(P4)$$

Ainsi, dans le but de résoudre P4, on résout P5 – il existe des méthodes efficaces pour le faire – et une solution optimale de P5, $x_1^*, \dots, x_n^*, y_1^*, \dots, y_l^*$ fournit directement une solution optimale de P4 : x_1^*, \dots, x_n^* .

$$(P5) \begin{cases} \min L(x_1, \dots, x_n, y_1, \dots, y_l) \\ \text{s.c.} \begin{cases} f_i(x, y) \leq 0 & i = 1, \dots, t & (5.1) \\ x_j \geq 0 & j = 1, \dots, r & (5.2) & y_j \geq 0 & j = 1, \dots, p & (5.5) \\ x_j \in \{0, 1\} & j = r + 1, \dots, s & (5.3) & y_j \in \{0, 1\} & j = p + 1, \dots, q & (5.6) \\ x_j \in N & j = s + 1, \dots, n & (5.4) & y_j \in N & j = q + 1, \dots, l & (5.7) \end{cases} \end{cases}$$

Les fonctions $L(x,y)$ et $f_i(x,y)$ ($i=1,\dots,t$) apparaissant dans P5 sont des fonctions linéaires. Les vecteurs $x=(x_1,\dots,x_n)$ et $y=(y_1,\dots,y_n)$ sont les vecteurs des inconnues.

Nous étudierons également certaines transformations de programmes mathématiques non linéaires en PQCVM, par exemple, celles qui associent le programme P6 au programme P4. Dans P6, $L(x,y)$ et $f_i(x,y)$ ($i=1,\dots,t$) sont des fonctions linéaires des variables x_j ($j=1,\dots,n$) et y_j ($j=1,\dots,l$), et $Q(x,y)$ est une fonction quadratique convexe des mêmes variables. Nous dirons que le programme P6 est une formulation de type PQCVM du programme P4 si P6 admet une solution optimale et si la propriété suivante est vérifiée :

$$(x^*,y^*) \in S_{\text{opt}}(\text{P6}) \Rightarrow x^* \in S_{\text{opt}}(\text{P4})$$

Nous considérerons essentiellement des instances de P6 où les variables y_j ($j=1,\dots,l$) ne sont pas présentes, c'est-à-dire des instances où la fonction économique est une fonction quadratique convexe des variables x_1,\dots,x_n de P4.

$$(\text{P6}) \begin{cases} \min L(x_1,\dots,x_n,y_1,\dots,y_l) + Q(x_1,\dots,x_n,y_1,\dots,y_l) \\ \text{s.c.} \begin{cases} f_i(x,y) \leq 0 & i=1,\dots,t & (6.1) \\ x_j \geq 0 & j=1,\dots,r & (6.2) & y_j \geq 0 & j=1,\dots,p & (6.5) \\ x_j \in \{0,1\} & j=r+1,\dots,s & (6.3) & y_j \in \{0,1\} & j=p+1,\dots,q & (6.6) \\ x_j \in N & j=s+1,\dots,n & (6.4) & y_j \in N & j=q+1,\dots,l & (6.7) \end{cases} \end{cases}$$

Essayer de résoudre un problème d'optimisation en le formulant comme un programme mathématique puis en le transformant en un PLVM ou un PQCVM n'est pas toujours une bonne approche, mais nous verrons que, dans de nombreux cas, celle-ci est très intéressante. Une des motivations importantes pour se ramener à ce type de programmes mathématiques est qu'il existe aujourd'hui de nombreux logiciels commerciaux, appelés quelquefois *solveurs*, fiables, extrêmement performants et faciles à utiliser pour résoudre ces programmes. On peut ainsi traiter des problèmes d'optimisation, sans trop d'effort et de façon aussi efficace – et même quelquefois plus efficace – qu'avec un algorithme spécifique, conçu spécialement pour le problème et souvent difficile à mettre en œuvre. Les problèmes d'optimisation en variables mixtes avec une fonction économique linéaire ou quadratique convexe sont en effet des problèmes bien connus, étudiés depuis longtemps et dont la résolution a fait des progrès considérables durant les quinze dernières années. Ainsi, grâce à ces logiciels et à la puissance des ordinateurs actuels on peut résoudre des programmes comportant des milliers de variables. Par ailleurs, il existe des langages de modélisation, appelés quelquefois *modeleurs*, qui apportent une aide très importante à la mise en œuvre de ces méthodes.

5. COMPLEXITÉ DES ALGORITHMES ET DES PROBLÈMES

Un algorithme peut être vu comme une méthode permettant de résoudre un problème sur un ordinateur. La complexité (en temps) d'un algorithme se mesure par son coût en temps de calcul en fonction de la taille de l'instance du problème considérée, c'est-à-dire par le nombre d'opérations élémentaires (additions, soustractions, comparaison, affectation, etc.) engendrées par l'exécution de l'algorithme, ce nombre étant exprimé en fonction de la taille de l'instance. Par exemple, le nombre d'opérations élémentaires engendrées par l'algorithme classique permettant de calculer le produit de deux matrices carrées de dimension n est inférieur ou égal à cn^3 où c est une constante suffisamment grande. Dans cet exemple, la taille de l'instance est égale à n^2 , le nombre de coefficients d'une matrice carrée d'ordre n . On dit que l'algorithme est polynomial et que sa complexité est en $O(n^3)$. Considérons un graphe de n sommets, non orienté, $G = (X, A)$ (X désigne l'ensemble des sommets et A , l'ensemble des arêtes) dont toutes les arêtes ont une valeur positive ou nulle. Soit s et t deux sommets particuliers de G . Le problème de la chaîne la plus courte de s à t consiste à trouver, parmi toutes les chaînes de G reliant s et t , une chaîne de valeur minimale, la valeur d'une chaîne étant égale à la somme des valeurs des arêtes qui la composent. De nombreux algorithmes polynomiaux permettent de résoudre ce problème. Il existe, par exemple, un algorithme dont le nombre d'opérations est inférieur à cn^2 où c est une constante suffisamment grande. Il s'agit donc d'un algorithme polynomial en $O(n^2)$. Considérons maintenant le problème de l'ensemble stable de cardinal maximal d'un graphe. Un sous-ensemble de sommets de G est un ensemble stable si toutes les paires de sommets de cet ensemble ne sont pas reliées par une arête. Le problème de l'ensemble stable de cardinal maximal consiste à déterminer, dans le graphe G , un ensemble stable comportant le plus grand nombre possible de sommets. On ne connaît pas d'algorithme polynomial pour résoudre ce problème et, de plus, il est conjecturé qu'il n'en existe pas. Autrement dit, il est conjecturé qu'il n'existe pas d'entiers c , p et d'algorithmes pour résoudre le problème, tels que le nombre d'opérations élémentaires engendrées par l'exécution de l'algorithme soit inférieur à cn^p , pour tout graphe de n sommets. Cette distinction entre polynomial et non polynomial est importante car *polynomial* est généralement synonyme d'*efficace* et non *polynomial*, de *non efficace*.

Intéressons-nous maintenant à des problèmes de décision, c'est-à-dire à des problèmes dont la solution est « oui » ou « non ». Voyons quelques exemples. Étant donné une valeur v , G admet-il une chaîne, reliant s à t , de valeur inférieure

$$(P8) \left\{ \begin{array}{l} \text{Max } f_L(y) = \sum_{(i,j) \in T} w_{ij} y_{ij} \\ \text{s.c. } \left\{ \begin{array}{l} \sum_{i=1}^n x_i = k \quad (8.1) \\ y_{ij} \leq x_i \quad (i,j) \in T \quad (8.2) \\ y_{ij} \leq x_j \quad (i,j) \in T \quad (8.3) \\ y_{ij} \geq 0 \quad (i,j) \in T \quad (8.4) \\ x_i \in \{0,1\} \quad i = 1, \dots, n \quad (8.5) \end{array} \right. \end{array} \right.$$

On peut vérifier que, dans toute solution optimale de P8, $y_{ij} = x_i x_j$ si le poids de l'arête $[v_i, v_j]$ est positif (voir chapitre 4). Montrons maintenant comment engendrer une instance du problème – correspondant à un graphe complet – et le programme P8 associé en utilisant le langage AMPL. On crée tout d'abord le fichier FP8 : les deux premières instructions attribuent des valeurs aux paramètres n et k ; elles définissent donc le nombre de sommets du graphe G et le nombre de sommets du sous-graphe optimal recherché. Dans cet exemple d'utilisation du langage AMPL, on considère un graphe complet de 20 sommets, et l'on cherche un sous-graphe de 10 sommets dont l'ensemble des arêtes est de poids maximal. Les deux instructions suivantes définissent deux ensembles d'indices, $I = \{1, \dots, 20\}$ et $J = \{1, \dots, 20\}$. L'instruction suivante permet de donner des poids aux arêtes de G . Ici, ces poids sont engendrés automatiquement, suivant la loi uniforme, dans l'intervalle $[0,100]$, puis arrondis à l'entier le plus proche. On obtient, par exemple, $w_{12} = 65, w_{13} = 4, w_{14} = 87$, etc. Il serait tout aussi facile de définir des coefficients w_{ij} à partir de poids donnés pour chaque arête. Les deux instructions suivantes définissent les variables du problème : les variables bivalentes x_i ($i = 1, \dots, n$) et les variables non négatives y_{ij} ($i = 1, \dots, n-1; j = i+1, \dots, n$). La ligne suivante indique que l'on cherche à maximiser la fonction $\sum_{i=1}^{n-1} \sum_{j=i+1}^n w_{ij} y_{ij}$. Enfin, les trois dernières lignes expriment, respectivement, les contraintes $\sum_{i=1}^n x_i = k, y_{ij} \leq x_i$ ($i = 1, \dots, n-1; j = i+1, \dots, n$) et $y_{ij} \leq x_j$ ($i = 1, \dots, n-1; j = i+1, \dots, n$).

```
# FP8 : Fichier AMPL correspondant
# au programme mathématique P8
param n:=20;
param k:=10;
set I:= 1..n;
set J:= 1..n;
param w{i in I,j in J}:=round(Uniform(0,100));
var x{i in I} binary;
var y{i in I,j in J:i<j}>=0;
maximize f : sum {i in I,j in J : i<j} w[i,j]*y[i,j];
subject to
C1: sum {i in I} x[i]=k;
C2 {i in I,j in J : i<j} : y[i,j]<=x[i];
C3 {i in I,j in J : i<j} : y[i,j]<=x[j];
```

l'usine implantée sur ce site. La production du site s_i est donnée par l'expression $\sum_{k=1}^m N_k x_{ki}$ qui comporte, au plus, un terme x_{ki} non nul, à cause des contraintes 10.2. On obtient donc $\sum_{j=1}^p y_{ij} = \sum_{k=1}^m N_k x_{ki}$ ($i = 1, \dots, n$). En résumé, la solution du problème est obtenue en résolvant le programme linéaire en variables entières P10 :

$$(P10) \left\{ \begin{array}{l} \min \sum_{k=1}^m \sum_{i=1}^n c_{ki} x_{ki} + \sum_{i=1}^n \sum_{j=1}^p C d_{ij} y_{ij} \\ \text{s.c.} \left\{ \begin{array}{ll} \sum_{i=1}^n x_{ki} = 1 & k = 1, \dots, m \quad (10.1) \\ \sum_{k=1}^m x_{ki} \leq 1 & i = 1, \dots, n \quad (10.2) \\ \sum_{i=1}^n y_{ij} = q_j & j = 1, \dots, p \quad (10.3) \\ \sum_{j=1}^p y_{ij} = \sum_{k=1}^m N_k x_{ki} & i = 1, \dots, n \quad (10.4) \\ x_{ki} \in \{0, 1\} & k = 1, \dots, m ; i = 1, \dots, n \quad (10.5) \\ y_{ij} \in N & i = 1, \dots, n ; j = 1, \dots, p \quad (10.6) \end{array} \right. \end{array} \right.$$



Application numérique. Considérons 3 usines ($m = 3$), 5 sites potentiels ($n = 5$) et 9 clients ($p = 9$). Supposons $C = 6$ €/article/km. L'usine U_1 produit 1 800 articles, l'usine U_2 produit 1 200 articles et l'usine U_3 produit 2 200 articles. La demande des clients $cl_1, cl_2, cl_3, cl_4, cl_5, cl_6, cl_7, cl_8, cl_9$ est égale, respectivement, à 550, 370, 540, 660, 500, 480, 750, 650, 700. Le tableau 1 donne le coût, en milliers d'euros, de fonctionnement des usines sur les différents sites et le tableau 2 donne les distances en kilomètres séparant chaque site de chaque client.

Tableau 1. Matrice des coefficients c_{ij} , coût de fonctionnement de l'usine U_i sur le site s_j ($i = 1, 2, 3 ; j = 1, 2, 3, 4, 5$)

	Site s_1	Site s_2	Site s_3	Site s_4	Site s_5
Usine U_1	2 000	2 500	3 000	2 000	2 800
Usine U_2	1 500	2 000	1 800	2 100	1 800
Usine U_3	2 300	3 000	2 500	1 800	2 600

Tableau 2. Matrice des coefficients d_{ij} , distance en km du site s_i au client cl_j ($i = 1, 2, 3, 4, 5$; $j = 1, 2, 3, 4, 5, 6, 7, 8, 9$)


	Client cl_1	Client cl_2	Client cl_3	Client cl_4	Client cl_5	Client cl_6	Client cl_7	Client cl_8	Client cl_9
Site s_1	250	250	200	450	650	700	500	600	600
Site s_2	300	200	550	750	750	600	350	350	200
Site s_3	500	300	300	300	350	350	350	450	550
Site s_4	650	400	350	200	200	350	450	550	700
Site s_5	700	400	700	700	450	200	150	150	400

En utilisant un logiciel de résolution de programmes linéaires en variables entières, on obtient la solution suivante : l'usine U_1 est implantée sur le site s_2 , l'usine U_2 est implantée sur le site s_5 et l'usine U_3 est implantée sur le site s_4 ; le coût de fonctionnement correspondant est égal à 6 100 K€. Le tableau 3 indique, pour chaque site sur lequel l'implantation d'une usine a été retenue, le nombre d'articles que ce site fournit à chaque client.

Tableau 3. Valeurs optimales des variables y_{ij} , quantité livrée au client cl_j à partir du site s_i ($j = 1, 2, 3, 4, 5, 6, 7, 8, 9$; $i = 2, 4, 5$)

	Client cl_1	Client cl_2	Client cl_3	Client cl_4	Client cl_5	Client cl_6	Client cl_7	Client cl_8	Client cl_9	Total
Site s_2 Usine U_1	550	370	0	0	0	0	180	0	700	1 800
Site s_4 Usine U_3	0	0	540	660	500	480	20	0	0	2 200
Site s_5 Usine U_2	0	0	0	0	0	0	550	650	0	1 200
Total	550	370	540	660	500	480	750	650	700	

Le coût total de livraison des articles aux différents clients est égal à 7 320 K€. Le coût total de cette solution optimale est donc égal à 13 420 K€.

 **Exemple 2.** Un problème d'optimisation se formulant directement par un programme quadratique convexe en variables mixtes : constitution d'un portefeuille optimal.

Propriété 2.1.1. Pour tout $x \in \{0,1\}$, pour tout $y \in [0, U(y)]$ et pour tout $e \in R$, $e = xy$ si et seulement si les contraintes S2.1.1 sont vérifiées :


$$(S2.1.1) \begin{cases} e \leq xU(y) & (S2.1.1.1) \\ e \leq y & (S2.1.1.2) \\ e \geq y - (1-x)U(y) & (S2.1.1.3) \\ e \geq 0 & (S2.1.1.4) \end{cases}$$

Preuve. Supposons $e = xy$ et examinons les deux valeurs possibles de la variable x . Si $x = 0$, alors $e = 0$ et le système d'inéquations S2.1.1 devient S2.1.2 qui est vérifié. Si $x = 1$, alors $e = y$ et le système d'inéquations S2.1.1 devient S2.1.3 qui est vérifié.

$$(S2.1.2) \begin{cases} 0 \leq 0 & (S2.1.2.1) \\ 0 \leq y & (S2.1.2.2) \\ 0 \geq y - U(y) & (S2.1.2.3) \\ 0 \geq 0 & (S2.1.2.4) \end{cases} \quad (S2.1.3) \begin{cases} y \leq U(y) & (S2.1.3.1) \\ y \leq y & (S2.1.3.2) \\ y \geq y & (S2.1.3.3) \\ y \geq 0 & (S2.1.3.4) \end{cases}$$

Supposons maintenant que le système d'inéquations S2.1.1 soit vérifié et examinons, de nouveau, les deux valeurs possibles de la variable x . Si $x = 0$, S2.1.1 devient S2.1.4 et l'on a bien $e = 0$. Si $x = 1$, alors le système d'inéquations S2.1.1 devient S2.1.5 et l'on a bien $e = y$.

$$(S2.1.4) \begin{cases} e \leq 0 & (S2.1.4.1) \\ e \leq y & (S2.1.4.2) \\ e \geq y - U(y) & (S2.1.4.3) \\ e \geq 0 & (S2.1.4.4) \end{cases} \quad (S2.1.5) \begin{cases} e \leq U(y) & (S2.1.5.1) \\ e \leq y & (S2.1.5.2) \\ e \geq y & (S2.1.5.3) \\ e \geq 0 & (S2.1.5.4) \end{cases} \quad \square$$

 **Exemple 2.1.1.** Considérons le programme mathématique P2.1.1 suivant. C'est un programme non linéaire puisque des produits de deux variables apparaissent à la fois dans la fonction économique et dans les contraintes.

$$(P2.1.1) \begin{cases} \max & 2x_1 + 5x_2 - x_4 + 2x_1x_4 \\ & \left| \begin{array}{l} x_2 + x_3 - 3x_4 + 3x_2x_3 \leq 2x_1x_4 & (2.1.1.1) \\ x_1 - x_2 + 2x_3 + 2x_2x_3 \geq 12 & (2.1.1.2) \end{array} \right. \\ \text{s.c.} & \left| \begin{array}{l} x_1, x_2 \in \{0,1\} & (2.1.1.3) \\ 0 \leq x_3 \leq 5 & (2.1.1.4) \\ 0 \leq x_4 \leq 8 & (2.1.1.5) \end{array} \right. \end{cases}$$

Remplaçons, dans P2.1.1, le produit x_1x_4 par la nouvelle variable e_{14} et le produit x_2x_3 , par la nouvelle variable e_{23} . Ajoutons les contraintes de linéarisation, fondées sur la propriété 2.1.1, afin de forcer la variable e_{14} à prendre la valeur du produit x_1x_4 et la variable e_{23} , celle du produit x_2x_3 . On obtient le programme linéaire en variables mixtes P2.1.2. Il est équivalent au programme non linéaire en variables mixtes P2.1.1 dans le sens suivant : $(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4)$ est une solution admissible de P2.1.1 de valeur v si et seulement s'il existe des valeurs \tilde{e}_{14} et \tilde{e}_{23} des variables e_{14} et e_{23} telles que $(\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4, \tilde{e}_{14}, \tilde{e}_{23})$ soit une solution admissible de P2.1.2 de valeur v . On obtiendra donc immédiatement une solution optimale de P2.1.1 à partir d'une solution optimale de P2.1.2. Ainsi, si $(x_1^*, x_2^*, x_3^*, x_4^*, e_{14}^*, e_{23}^*)$ est une solution optimale de P2.1.2, alors $(x_1^*, x_2^*, x_3^*, x_4^*)$ est une solution optimale de P2.1.1.

$$(P2.1.2) \begin{cases} \max & 2x_1 + 5x_2 - x_4 + 2e_{14} \\ & x_2 + x_3 - 3x_4 + 3e_{23} \leq 2e_{14} & (2.1.2.1) \\ & x_1 - x_2 + 2x_3 + 2e_{23} \geq 12 & (2.1.2.2) \\ \text{s.c.} & e_{14} \leq 8x_1; e_{14} \leq x_4; e_{14} \geq x_4 - 8(1 - x_1); e_{14} \geq 0 & (2.1.2.3) \\ & e_{23} \leq 5x_2; e_{23} \leq x_3; e_{23} \geq x_3 - 5(1 - x_2); e_{23} \geq 0 & (2.1.2.4) \\ & x_1, x_2 \in \{0, 1\}; 0 \leq x_3 \leq 5; 0 \leq x_4 \leq 8 & (2.1.2.5) \end{cases}$$

Une solution optimale de P2.1.2 est $x_1 = 1, x_2 = 1, x_3 = 3, x_4 = 8, e_{14} = 8, e_{23} = 3$. Elle donne à la fonction économique la valeur 15. Une solution optimale de P2.1.1 est donc $x_1 = 1, x_2 = 1, x_3 = 3, x_4 = 8$.

2.2. Transformation d'une variable entière en une fonction linéaire de variables bivalentes

En utilisant l'écriture des nombres entiers en base 2, il n'est pas difficile d'exprimer une variable entière comme une fonction linéaire de variables bivalentes. Considérons une variable entière x ne pouvant prendre que des valeurs comprises entre 0 et $U(x)$. On peut considérer, à la place de la variable x , l'expression linéaire de variables bivalentes $\sum_{i=1}^p 2^{i-1} t_i$ avec $p = \lceil \log_2(U(x) + 1) \rceil$ et imposer à cette expression d'être inférieure ou égale à $U(x)$. À partir de cette réécriture d'une variable entière, on peut linéariser un certain nombre d'expressions non linéaires faisant intervenir des variables entières. Par exemple, en utilisant la propriété 2.1.1 du paragraphe précédent, on peut linéariser le produit d'une variable entière x inférieure ou égale à $U(x)$ par une variable quelconque y inférieure ou égale à $U(y)$ et notamment le produit d'une variable entière par une variable réelle. On peut ainsi

4.2. Produit de n variables bivalentes

$$\prod_{i=1}^n x_i$$

$$x_i \in \{0,1\} \quad (i = 1, \dots, n)$$

Pour tout $x \in \{0,1\}^n$, pour tout $e \in \mathbb{R}$,
 $e = \prod_{i=1}^n x_i$ ssi :

Solution n° 1 : les contraintes [C1] ci-contre sont vérifiées.

Solution n° 2 : les contraintes [C2] ci-contre sont vérifiées.

$$(C1) \begin{cases} e \leq x_i & (\forall i) \\ e \geq \sum_{i=1}^n x_i - n + 1 \\ e \geq 0 \end{cases}$$

$$(C2) \begin{cases} e \leq (1/n) \sum_{i=1}^n x_i \\ e \geq \sum_{i=1}^n x_i - n + 1 \\ e \in \{0,1\} \end{cases}$$

Exemple (solution n° 1)

Problème non linéaire	Problème linéaire
$\left\{ \begin{array}{l} \max \quad 4x_1x_2x_3 + 3x_2x_3x_4 - 6x_1x_2x_3x_4 \\ \text{s.c.} \quad \begin{cases} 4x_1x_2x_3 - 2x_2x_3x_4 \leq 3 \\ x_1x_3x_4 + x_2x_3x_4 \leq 2 \\ x_1, x_2, x_3, x_4 \in \{0,1\} \end{cases} \end{array} \right.$	$\left\{ \begin{array}{l} \max \quad 4y_{123} + 3y_{234} - 6y_{1234} \\ \text{s.c.} \quad \begin{cases} 4y_{123} - 2y_{234} \leq 3 \\ y_{134} + y_{234} \leq 2 \\ y_{123} \leq x_1 ; y_{123} \leq x_2 ; y_{123} \leq x_3 \\ y_{123} \geq x_1 + x_2 + x_3 - 2 \\ y_{134} \leq x_1 ; y_{134} \leq x_3 ; y_{134} \leq x_4 \\ y_{134} \geq x_1 + x_3 + x_4 - 2 \\ y_{234} \leq x_2 ; y_{234} \leq x_3 ; y_{234} \leq x_4 \\ y_{234} \geq x_2 + x_3 + x_4 - 2 \\ y_{1234} \leq x_1 ; y_{1234} \leq x_2 ; y_{1234} \leq x_3 ; y_{1234} \leq x_4 \\ y_{1234} \geq x_1 + x_2 + x_3 + x_4 - 3 \\ y_{123}, y_{134}, y_{234}, y_{1234} \geq 0 \\ x_1, x_2, x_3, x_4 \in \{0,1\} \end{cases} \end{array} \right.$
<p>Solution optimale : $x_1 = 0, x_2 = 1, x_3 = 1, x_4 = 1$ ($y_{123} = 0, y_{134} = 0, y_{234} = 1, y_{1234} = 0$) ;</p> <p>Valeur de la solution optimale : 3.</p>	

4.3. Produit d'une variable bivalente par une variable réelle non négative et bornée

$$x y$$

$$x \in \{0,1\}, y \in [0, U(y)]$$

Pour tout $x \in \{0,1\}$, pour tout $y \in [0, U(y)]$ et pour tout $e \in \mathbb{R}$, $e = xy$ ssi les contraintes ci-contre sont vérifiées.

$$\begin{cases} e \leq U(y)x \\ e \leq y \\ e \geq y - U(y)(1-x) \\ e \geq 0 \end{cases}$$

Exemple

Problème non linéaire	Problème linéaire
$\begin{cases} \min & 4x_1y_1 + 3x_1y_2 - 3x_2y_2 + 2x_1x_2 \\ \text{s.c.} & \begin{cases} x_1 - y_2 + x_2y_1 \geq 6 \\ y_1 + y_2 \leq 10 \\ y_1, y_2 \geq 0 \\ x_1, x_2 \in \{0,1\} \end{cases} \end{cases}$	$\begin{cases} \min & 4z_{11} + 3z_{12} - 3z_{22} + 2t_{12} \\ \text{s.c.} & \begin{cases} x_1 - y_2 + z_{21} \geq 6 \\ y_1 + y_2 \leq 10 \\ z_{11} \leq 10x_1; z_{11} \leq y_1; z_{11} \geq y_1 - 10(1-x_1) \\ z_{12} \leq 10x_1; z_{12} \leq y_2; z_{12} \geq y_2 - 10(1-x_1) \\ z_{21} \leq 10x_2; z_{21} \leq y_1; z_{21} \geq y_1 - 10(1-x_2) \\ z_{22} \leq 10x_2; z_{22} \leq y_2; z_{22} \geq y_2 - 10(1-x_2) \\ t_{12} \leq x_1; t_{12} \leq x_2; 1 - x_1 - x_2 + t_{12} \geq 0 \\ z_{11}, z_{12}, z_{21}, z_{22}, t_{12} \geq 0 \\ y_1, y_2 \geq 0 \\ x_1, x_2 \in \{0,1\} \end{cases} \end{cases}$
<p>Commentaire : la deuxième contrainte du problème non linéaire implique $y_1 \leq 10$ et $y_2 \leq 10$; les produits $x_1y_1, x_1y_2, x_2y_1, x_2y_2, x_1x_2$ sont remplacés par les variables $z_{11}, z_{12}, z_{21}, z_{22}, t_{12}$.</p> <p>Solution optimale : $x_1 = 0, x_2 = 1, y_1 = 8, y_2 = 2$ ($z_{11} = 0, z_{12} = 0, z_{21} = 8, z_{22} = 2, t_{12} = 0$) ;</p> <p>Valeur de la solution optimale : -6.</p>	

4.4. Produit d'une variable entière non négative et bornée par une variable réelle non négative et bornée

$$\begin{matrix} xy \\ x \in \{0, 1, \dots, U(x)\}, y \in [0, U(y)] \end{matrix}$$

Soit $p = \lceil \log_2(U(x) + 1) \rceil$.

Pour tout $x \in \{0, 1, \dots, U(x)\}$, pour tout $y \in [0, U(y)]$ et pour tout $e \in \mathbb{R}$, $e = xy$ ssi il existe $t \in \mathbb{R}^{+p}$ et $z \in \{0, 1\}^p$ t.q. les contraintes ci-contre soient vérifiées.

$$\begin{cases} x = \sum_{i=1}^p 2^{i-1} z_i \\ e = \sum_{i=1}^p 2^{i-1} t_i \\ t_i \leq z_i U(y) & (\forall i) \\ t_i \leq y & (\forall i) \\ t_i \geq y - U(y)(1 - z_i) & (\forall i) \end{cases}$$

Commentaire : $t_i = z_i y$

Exemple

Problème non linéaire	Problème linéaire
$\begin{cases} \max & 3xy \\ \text{s.c.} & \begin{cases} 2x + 3y \leq 14 \\ x \in \mathbb{N} \\ y \geq 0 \end{cases} \end{cases} \rightarrow$	$\begin{cases} \max & 3e \\ \text{s.c.} & \begin{cases} 2x + 3y \leq 14 \\ x = z_1 + 2z_2 + 4z_3 \\ e = t_1 + 2t_2 + 4t_3 \\ t_1 \leq 5z_1; t_1 \leq y; t_1 \geq y - 5(1 - z_1) \\ t_2 \leq 5z_2; t_2 \leq y; t_2 \geq y - 5(1 - z_2) \\ t_3 \leq 5z_3; t_3 \leq y; t_3 \geq y - 5(1 - z_3) \\ y, t_1, t_2, t_3 \geq 0; z_1, z_2, z_3 \in \{0, 1\} \end{cases} \end{cases}$
<p>Solution optimale : $x = 3, y = 8/3$ [$e = 8, z_1 = 1, z_2 = 1, z_3 = 0, t_1 = 8/3, t_2 = 8/3, t_3 = 0$];</p> <p>Valeur de la solution optimale : 24.</p>	

4.5. Produit d'une fonction linéaire de variables bivalentes par une fonction linéaire de variables réelles non négatives

$$f(x) \times g(y)$$

$$f(x) = \sum_{i=1}^m a_i x_i, g(y) = \sum_{i=1}^n b_i y_i,$$

$$x_i \in \{0,1\} \ (i=1,\dots,m), \ y_i \geq 0 \ (i=1,\dots,n)$$

$$a_i \geq 0 \ (i=1,\dots,m), \ b_i \geq 0 \ (i=1,\dots,n),$$

$$\sum_{i=1}^n b_i y_i \leq U(g)$$

Pour tout $x \in \{0,1\}^m$, pour tout $y \in \mathbb{R}^{+n}$ t.q.
 $\sum_{i=1}^n b_i y_i \leq U(g)$ et pour tout $e \in \mathbb{R}$,
 $e = (\sum_{i=1}^m a_i x_i) \times (\sum_{i=1}^n b_i y_i)$ ssi il existe $g \in \mathbb{R}$
 et $t \in \mathbb{R}^{+n}$ t.q. les contraintes ci-contre soient
 vérifiées.

$$\begin{cases} e = \sum_{i=1}^m a_i t_i \\ g = \sum_{i=1}^n b_i y_i \\ t_i \leq x_i U(g) & (\forall i) \\ t_i \leq g & (\forall i) \\ t_i \geq g - U(g)(1 - x_i) & (\forall i) \end{cases}$$

Commentaire : la variable réelle et non négative t_i représente le produit de la variable bivalente x_i par la variable g qui représente la fonction $g(y)$.

Exemple

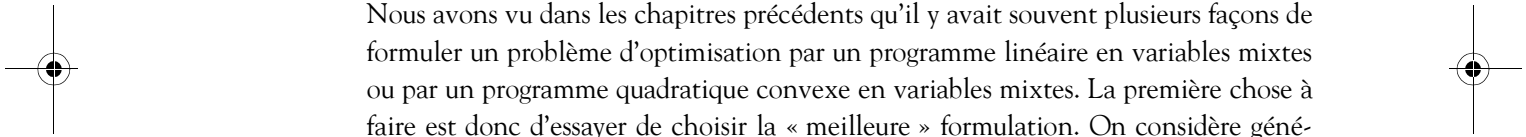
Problème non linéaire	Problème linéaire
$\begin{cases} \max & (4x_1 + 3x_2 + x_3)(2y_1 + 3y_2 + 4y_3) \\ \text{s.c.} & \begin{cases} 3x_1 - x_2 + 5x_3 + 2y_1 - y_2 + 4y_3 \leq 12 \\ y_2 \leq 6 \\ x_1, x_2, x_3 \in \{0,1\} \\ y_1, y_2, y_3 \geq 0 \end{cases} \end{cases}$	$\begin{cases} \max & e \\ \text{s.c.} & \begin{cases} e = 4t_1 + 3t_2 + t_3; g = 2y_1 + 3y_2 + 4y_3 \\ t_1 \leq 56x_1; t_1 \leq g; t_1 \geq g - 56(1 - x_1) \\ t_2 \leq 56x_2; t_2 \leq g; t_2 \geq g - 56(1 - x_2) \\ t_3 \leq 56x_3; t_3 \leq g; t_3 \geq g - 56(1 - x_3) \\ 3x_1 - x_2 + 5x_3 + 2y_1 - y_2 + 4y_3 \leq 12 \\ y_1, y_2, y_3, t_1, t_2, t_3 \geq 0; x_1, x_2, x_3 \in \{0,1\}; y_2 \leq 6 \end{cases} \end{cases}$
<p>Commentaire : toutes les variables étant positives ou nulles, on peut déduire des trois contraintes $3x_1 - x_2 + 5x_3 + 2y_1 - y_2 + 4y_3 \leq 12$, $x_2 \leq 1$ et $y_2 \leq 6$, les trois inégalités $2y_1 \leq 19$, $3y_2 \leq 18$ et $4y_3 \leq 19$. On en déduit donc : $2y_1 + 3y_2 + 4y_3 \leq 56$.</p> <p>Solution optimale : $x_1 = 1, x_2 = 1, x_3 = 0, y_1 = 8, y_2 = 6, y_3 = 0$ ($e = 238, g = 34, t_1 = 34, t_2 = 34, t_3 = 0$);</p> <p>Valeur de la solution optimale : 238.</p>	



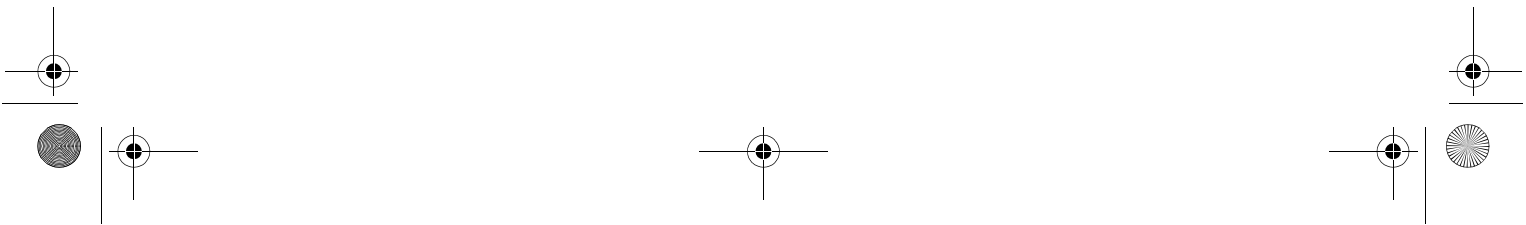
6

Pré-traitements

1. INTRODUCTION



Nous avons vu dans les chapitres précédents qu'il y avait souvent plusieurs façons de formuler un problème d'optimisation par un programme linéaire en variables mixtes ou par un programme quadratique convexe en variables mixtes. La première chose à faire est donc d'essayer de choisir la « meilleure » formulation. On considère généralement que, pour un problème P , une formulation est meilleure qu'une autre si elle permet de résoudre plus rapidement, en moyenne, un ensemble d'instances tests du problème P ou, de façon équivalente, si elle permet de résoudre, en moyenne, des instances de P de plus grande taille dans un temps donné. Si l'on s'intéresse à une instance particulière du problème P , on considère que la formulation F est meilleure que la formulation F' si le problème est résolu plus rapidement à l'aide de la formulation F ou encore, si la formulation F permet de résoudre le problème dans le temps imparti alors que la formulation F' ne le permet pas. Dans ce cas, l'étude du comportement expérimental moyen des deux formulations n'est pas intéressante. Nous avons vu au chapitre 3 quelques pistes pour choisir une bonne formulation. Lorsqu'on a retenu une formulation d'un problème d'optimisation par un PLVM ou un PQCVM on peut encore essayer d'améliorer cette formulation en effectuant certains pré-traitements. Ces pré-traitements sont des opérations plus ou moins complexes qui sont effectuées à partir de la formulation choisie, pour l'améliorer. L'amélioration peut consister, par exemple, à simplifier le problème, à déterminer ou affiner des coefficients, à fixer la valeur de certaines variables, à améliorer les bornes sur les variables, à ajouter de nouvelles contraintes, redondantes pour le problème initial mais non redondantes pour sa relaxation continue, etc. Le pré-traitement est donc une étape importante de l'étude du problème, qui se situe entre sa formulation




raisonnement en effectuant un peu d'énumération sur les variables. On résout, par exemple, les problèmes $(\overline{P3.1.2_{ij}})$ et $(\overline{P3.1.2_{i\bar{j}}})$:

$$(\overline{P3.1.2_{ij}}) \begin{cases} \max f(x_1, \dots, x_n) \\ x_i = 1 & (\overline{3.1.2_{ij}.1}) \\ x_j = 1 & (\overline{3.1.2_{ij}.2}) \\ x \in D' \supseteq D & (\overline{3.1.2_{ij}.3}) \end{cases}$$

$$(\overline{P3.1.2_{i\bar{j}}}) \begin{cases} \max f(x_1, \dots, x_n) \\ x_i = 1 & (\overline{3.1.2_{i\bar{j}}.1}) \\ x_j = 0 & (\overline{3.1.2_{i\bar{j}}.2}) \\ x \in D' \supseteq D & (\overline{3.1.2_{i\bar{j}}.3}) \end{cases}$$

Si les valeurs optimales de $(\overline{P3.1.2_{ij}})$ et $(\overline{P3.1.2_{i\bar{j}}})$ sont *toutes les deux* inférieures à $f(\tilde{x})$, alors on peut fixer la variable x_i à la valeur 0 pour la recherche de l'optimum de P3.1.1. Ce type de pré-traitement est efficace si l'on est capable de déterminer à la fois une bonne borne supérieure et une bonne solution admissible de P3.1.1. On peut alors obtenir une forte réduction de la taille du problème. Dans le cas de la borne supérieure, il faut, en fait, savoir calculer une bonne borne supérieure de P3.1.1 après avoir fixé la valeur de certaines variables

 **Application.** Considérons le problème de l'ensemble stable de poids maximal. Rappelons que, dans un graphe, un sous-ensemble de sommets S est un ensemble stable si toutes les paires de sommets de cet ensemble ne sont pas reliées par une arête et que le poids d'un ensemble stable est égal à la somme des poids de ses sommets. F3.1.1 est une formulation de ce problème.

 **Formulation mathématique.**

Soit le graphe $G = (V, E)$ où $V = \{v_1, \dots, v_n\}$ désigne l'ensemble des sommets et E , l'ensemble des arêtes.

À chaque sommet v_i du graphe G est associé un poids p_i positif ou nul.

❖ Déterminer un sous-ensemble S de V , de telle façon que :

- toute paire de sommets de S ne soit pas reliée par une arête ;
- la somme des poids des sommets de S soit maximale.

F3.1.1. Ensemble stable de poids maximal

Cherchons maintenant à déterminer un ensemble stable de poids maximal dans le graphe $G = (V, E)$ formé des 7 sommets $v_1, v_2, v_3, v_4, v_5, v_6, v_7$ de poids respectifs 5, 16, 15, 4, 4, 14, 18, et des 11 arêtes $[v_1, v_2], [v_1, v_3], [v_1, v_4], [v_2, v_4], [v_2, v_5], [v_2, v_6], [v_3, v_7], [v_4, v_5], [v_4, v_6], [v_5, v_6], [v_6, v_7]$. Déterminons un ensemble stable de ce graphe par l'heuristique suivante : choisir le sommet de plus fort poids, éliminer les sommets adjacents au sommet choisi, choisir de nouveau le sommet de plus fort poids dans le graphe obtenu, éliminer les sommets adjacents au sommet choisi, etc. Cette méthode, appliquée au graphe G , fournit l'ensemble stable de G formé des 2 sommets v_2 et v_7 ; son poids est égal à 34. Formulons le problème de l'ensemble stable de poids maximal de G par le PLVB P3.1.3 :

$$(P3.1.3) \begin{cases} \max \sum_{i=1}^7 p_i x_i \\ \text{s.c.} \begin{cases} x_i + x_j \leq 1 & [v_i, v_j] \in E & (3.1.3.1) \\ x_i \in \{0, 1\} & i = 1, \dots, 7 & (3.1.3.2) \end{cases} \end{cases}$$


La variable x_i est une variable bivalente qui vaut 1 si et seulement si le sommet v_i est retenu dans l'ensemble stable. Le coefficient p_i est le poids du sommet v_i . Considérons maintenant les deux programmes linéaires continus P3.1.4 et P3.1.5 obtenus en ajoutant à la relaxation continue de P3.1.3 les contraintes $x_3 = 1, x_2 = 1$ pour former P3.1.4, et les contraintes $x_3 = 1, x_2 = 0$ pour former P3.1.5. Notons que, le graphe considéré ne comportant pas de sommets isolés, les contraintes $x_i \leq 1$ ($i = 1, \dots, 7$) sont inutiles. Elles sont en effet impliquées par les contraintes $x_i + x_j \leq 1$ ($[v_i, v_j] \in E$).

$$(P3.1.4) \begin{cases} \max \sum_{i=1}^7 p_i x_i \\ \text{s.c.} \begin{cases} x_i + x_j \leq 1 & [v_i, v_j] \in E & (3.1.4.1) \\ 0 \leq x_i \leq 1 & i = 1, \dots, 7 & (3.1.4.2) \\ x_3 = 1 & & (3.1.4.3) \\ x_2 = 1 & & (3.1.4.4) \end{cases} \end{cases}$$

$$(P3.1.5) \begin{cases} \max \sum_{i=1}^7 p_i x_i \\ \text{s.c.} \begin{cases} x_i + x_j \leq 1 & [v_i, v_j] \in E & (3.1.5.1) \\ 0 \leq x_i \leq 1 & i = 1, \dots, 7 & (3.1.5.2) \\ x_3 = 1 & & (3.1.5.3) \\ x_2 = 0 & & (3.1.5.4) \end{cases} \end{cases}$$

la résolution de P3.2.1 par un logiciel de PLVB jusqu'à l'obtention d'une première solution admissible (branch and bound tronqué). Soit (\tilde{x}, \tilde{y}) une telle solution. Supposons que $\sum_{i=1}^n \tilde{x}_i = \alpha$. Cela signifie que l'on a trouvé un graphe comportant un ensemble stable de cardinal α . On peut alors fixer à 1 les α premières variables x_i et à 0, les variables y_{ij} telles que $1 \leq i < j \leq \alpha$. En effet, le fait de savoir qu'il existe un graphe respectant les contraintes et admettant un ensemble stable de cardinal α permet de rechercher un graphe extrémal comportant un ensemble stable de cardinal supérieur ou égal à α . On peut donc décider, sans perte de généralité, que les sommets v_1, \dots, v_α forment une partie de l'ensemble stable de cardinal maximal du graphe extrémal cherché, et fixer ainsi les α premières variables à 1. Par voie de conséquence, il ne doit pas exister d'arêtes entre deux sommets de cet ensemble stable et donc $y_{ij} = 0$ pour tous les couples (i, j) tels que $1 \leq i < j \leq \alpha$. On obtient ainsi le problème P3.2.2 qui, grâce aux fixations de variables, est généralement beaucoup plus facile à résoudre que P3.2.1.

$$\begin{array}{l}
 \text{(P3.2.2)} \left\{ \begin{array}{ll}
 \max \sum_{i=1}^n x_i & \\
 \left. \begin{array}{l}
 x_i + x_j \leq 2 - y_{ij} \quad 1 \leq i < j \leq n \quad (3.2.2.1) \\
 \sum_{i=1}^{n-1} \sum_{j=i+1}^n y_{ij} = m \quad (3.2.2.2) \\
 \sum_{j=1}^{i-1} y_{ji} + \sum_{j=i+1}^n y_{ij} \geq d_{\min} \quad 1 \leq i \leq n \quad (3.2.2.3) \\
 \sum_{j=1}^{i-1} y_{ji} + \sum_{j=i+1}^n y_{ij} \leq d_{\max} \quad 1 \leq i \leq n \quad (3.2.2.4) \\
 x_i = 1 \quad 1 \leq i \leq \alpha \quad (3.2.2.5) \\
 y_{ij} = 0 \quad 1 \leq i < j \leq \alpha \quad (3.2.2.6) \\
 x_i \in \{0, 1\} \quad 1 \leq i \leq n \quad (3.2.2.7) \\
 y_{ij} \in \{0, 1\} \quad 1 \leq i < j \leq n \quad (3.2.2.8)
 \end{array} \right. \\
 \text{s.c.}
 \end{array}
 \right.
 \end{array}$$

 **Expérimentation.** Cherchons le graphe admettant le plus grand ensemble stable possible et ayant les caractéristiques suivantes : $n = 50$, $m = 280$, $d_{\max} = 37$ et $d_{\min} = 10$. En lançant la résolution du programme P3.2.1 associé à cette instance, on obtient une première solution admissible de valeur 28 après 20 secondes de calcul (sur un PC muni d'un processeur Pentium IV à 1,8 GHz). En lançant P3.2.2 avec $\alpha = 28$, on obtient une solution optimale de P3.2.2 et donc de P3.2.1, de valeur 28, en une seconde de calcul. On a donc trouvé le graphe extrémal cherché en 21 secondes au total et son nombre de stabilité est égal à 28. Si l'on essaye de résoudre directement le programme P3.2.1, sans pré-traitement, 1 heure de calcul ne suffit pas pour obtenir la solution optimale. On voit que, dans ce cas, le pré-traitement proposé est particulièrement efficace.

de mauvaise qualité. La valeur optimale de la relaxation continue de P7.5 est souvent de bonne qualité mais ce problème comporte beaucoup de variables et de contraintes. Le programme P7.7 combine les deux avantages : qualité de la relaxation continue à la racine de l'arbre de recherche – au moins aussi bonne que celle de P7.5 – et petite taille.

Tableau 7.1. deux algorithmes de résolution du programme P7.1

	Algorithme A	Algorithme B
PRÉ-TRAITEMENT	Résoudre $\overline{P7.2}$, la relaxation continue de P7.2 ;	Résoudre $\overline{P7.5}$, la relaxation continue de P7.5 ;
	Soit (x^*, w^*, s^*) une solution optimale de $\overline{P7.2}$ et $f_L(x^*, w^*, s^*)$ sa valeur ;	Soit $(\hat{x}, \hat{w}, \hat{s})$ une solution optimale de $\overline{P7.5}$ et $f_L(\hat{x}, \hat{w}, \hat{s})$ sa valeur ;
	Construire à l'aide de cette solution la réécriture de la fonction $f(x)$: $f_L(x^*, w^*, s^*) + \sum_{i=1}^n q_i x_i$ $+ \sum_{i=1}^n \sum_{j=i+1}^n q_{ij} x_i x_j ;$	Construire à l'aide de cette solution la réécriture de la fonction $f(x)$: $f_L(\hat{x}, \hat{w}, \hat{s}) + \sum_{i=1}^n q'_i x_i$ $+ \sum_{i=1}^n \sum_{j=i+1}^n q'_{ij} x_i x_j$
	Considérer la forme réduite, P7.3, de P7.1 ;	Considérer la forme réduite, P7.6, de P7.1 ;
	Construire la linéarisation compacte, P7.4, de P7.3 ;	Construire la linéarisation compacte, P7.7, de P7.6 ;
RÉSOLUTION	Résoudre P7.4 par un solveur de PLVM.	Résoudre P7.7 par un solveur de PLVM.

Le type de réduction que nous venons de voir se généralise au problème de l'optimisation d'une fonction quadratique de variables 0-1 soumise à des contraintes linéaires quelconques.



Application numérique. Illustrons l'algorithme A sur le problème d'optimisation quadratique en variables bivalentes P7.8 :

$$(P7.8) \left\{ \begin{array}{l} \min f(x) = x_1 + 5x_2 + 2x_3 - 5x_4 - 4x_5 - 6x_6 \\ -3x_1x_4 + 2x_1x_5 - 4x_1x_6 - 5x_2x_4 + 6x_2x_5 + 5x_2x_6 \\ + 7x_3x_4 - 2x_3x_5 - x_3x_6 \\ \text{s.c.} \left\{ \begin{array}{l} x_1 + x_2 + x_3 = 1 \quad (7.8.1) \\ x_4 + x_5 + x_6 = 1 \quad (7.8.2) \\ x_1, \dots, x_6 \in \{0,1\} \quad (7.8.3) \end{array} \right. \end{array} \right.$$

Considérons la linéarisation P7.9 de P7.8 :

$$(P7.9) \left\{ \begin{array}{l} \min f_L(x, w) = x_1 + 5x_2 + 2x_3 - 5x_4 - 4x_5 - 6x_6 \\ -3w_{14} + 2w_{15} - 4w_{16} - 5w_{24} + 6w_{25} + 5w_{26} \\ + 7w_{34} - 2w_{35} - w_{36} \\ \text{s.c.} \left\{ \begin{array}{l} x_1 + x_2 + x_3 = 1 \quad (7.9.1) \\ x_4 + x_5 + x_6 = 1 \quad (7.9.2) \\ w_{ij} \leq x_i \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.9.3) \\ w_{ij} \leq x_j \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.9.4) \\ 1 - x_i - x_j + w_{ij} \geq 0 \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.9.5) \\ w_{ij} \geq 0 \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.9.6) \\ x_1, \dots, x_6 \in \{0,1\} \quad (7.9.7) \end{array} \right. \end{array} \right.$$

Considérons P7.10, la relaxation continue – écrite sous sa forme standard – de la linéarisation P7.9, c'est-à-dire le programme (P7.2) correspondant à cette instance :

$$(P7.10) \left\{ \begin{array}{l} \min f_L(x, w, s) = x_1 + 5x_2 + 2x_3 - 5x_4 - 4x_5 - 6x_6 \\ -3w_{14} + 2w_{15} - 4w_{16} - 5w_{24} + 6w_{25} + 5w_{26} \\ + 7w_{34} - 2w_{35} - w_{36} \\ \text{s.c.} \left\{ \begin{array}{l} x_1 + x_2 + x_3 = 1 \quad (7.10.1) \\ x_4 + x_5 + x_6 = 1 \quad (7.10.2) \\ w_{ij} + s_{ij}^1 = x_i \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.10.3) \\ w_{ij} + s_{ij}^2 = x_j \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.10.4) \\ 1 - x_i - x_j + w_{ij} - s_{ij}^3 = 0 \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.10.5) \\ w_{ij}, s_{ij}^1, s_{ij}^2, s_{ij}^3 \geq 0 \quad i \in \{1,2,3\}; j \in \{4,5,6\} \quad (7.10.6) \\ x_1, \dots, x_6 \geq 0 \quad (7.10.7) \end{array} \right. \end{array} \right.$$

La résolution de P7.10 par l'algorithme du simplexe permet d'écrire, pour toute solution admissible (x, w, s) de P7.9 :

$$\begin{aligned} f_L(x, w, s) &= -9 + x_4 + 4x_5 \\ &+ 2w_{15} + 6w_{25} + 5w_{26} + 7w_{34} \\ &+ 4s_{24}^1 + s_{36}^1 \\ &+ 3s_{14}^2 + 4s_{16}^2 + s_{24}^2 + 2s_{35}^2 \end{aligned}$$

soit, finalement, en remplaçant w_{ij} par $x_i x_j$, s_{ij}^1 par $x_i(1-x_j)$ et s_{ij}^2 par $(1-x_i)x_j$:

$$\begin{aligned} f(x) &= -9 + x_4 + 4x_5 \\ &+ 2x_1x_5 + 6x_2x_5 + 5x_2x_6 + 7x_3x_4 \\ &+ 4x_2(1-x_4) + x_3(1-x_6) \\ &+ 3(1-x_1)x_4 + 4(1-x_1)x_6 + (1-x_2)x_4 + 2(1-x_3)x_5 \end{aligned}$$

pour toute solution admissible de P7.8. D'après les contraintes 7.8.1 et 7.8.2, $1-x_1 = x_2 + x_3$, $1-x_2 = x_1 + x_3$, $1-x_3 = x_1 + x_2$, $1-x_4 = x_5 + x_6$ et $1-x_6 = x_4 + x_5$. En effectuant ces substitutions, on obtient la réduction suivante du problème P7.8 :

$$(P7.11) \left\{ \begin{array}{l} \min \quad -9 + x_4 + 4x_5 \\ \quad + x_1x_4 + 4x_1x_5 + 3x_2x_4 + 12x_2x_5 \\ \quad + 13x_2x_6 + 12x_3x_4 + x_3x_5 + 4x_3x_6 \\ \text{s.c.} \quad \left| \begin{array}{l} x_1 + x_2 + x_3 = 1 \quad (7.11.1) \\ x_4 + x_5 + x_6 = 1 \quad (7.11.2) \\ x_1, \dots, x_6 \in \{0,1\} \quad (7.11.3) \end{array} \right. \end{array} \right.$$

Écrivons maintenant la linéarisation de P7.11, de type P7.4 :

$$(P7.12) \left\{ \begin{array}{l} \min \quad -9 + x_4 + 4x_5 + z_1 + z_2 + z_3 \\ \quad (7.11.1), (7.11.2), (7.11.3) \\ \text{s.c.} \quad \left| \begin{array}{l} z_1 \geq x_4 + 4x_5 - 5(1-x_1) \quad (7.12.1) \\ z_2 \geq 3x_4 + 12x_5 + 13x_6 - 28(1-x_2) \quad (7.12.2) \\ z_3 \geq 12x_4 + x_5 + 4x_6 - 17(1-x_3) \quad (7.12.3) \\ z_1, z_2, z_3, z_4, z_5 \geq 0 \quad (7.12.3) \end{array} \right. \end{array} \right.$$

Références

- ACHER J., *Algèbre linéaire et programmation linéaire*. Dunod, 1965.
- ADAMS W. P. and FORRESTER R. J., *A Simple recipe for concise mixed 0-1 linearizations*. Operations Research Letters, 33, 2005, pp. 55-61.
- ADAMS W. P., FORRESTER R. J. and GLOVER F. W., *Comparison and enhancement strategies for linearizing mixed 0-1 quadratic programs*. Discrete Optimization, 1, 2004, pp. 99-120.
- ALJ A. et FAURE R., *Guide de la recherche opérationnelle*. Masson, 2 tomes, 1990.
- APPLEGET J. A and WOOD R. K., *Explicit-constraint branching for solving mixed-integer programs*. Computing tools for modeling, optimization and simulation, M. Laguna and J.L. Gonzalez-Velarde Ed., chapitre 14. Kluwer Academic Publishers, Boston, 2000.
- ARBEL A., *Exploring interior-point linear programming, algorithms and software*. MIT Press, 1993.
- BAZARAA M. S., JARVIS J. J and SHERALI H.D., *Linear programming and network flows*. Wiley, 2005.
- BEASLEY J. (Ed.), *Advances in linear and integer programming*. Oxford University Press, 1996.
- BERGE C., *Graphes et hypergraphes*. Dunod, 1970.
- BILLIONNET A., *Integer programming to schedule a hierarchical workforce with variable demands*. European Journal of Operational Research, 114, 1999, pp. 105-114.
- BILLIONNET A., *Approximate and exact solution methods for the hyperbolic 0-1 knapsack problem*. Information Systems and Operational Research, 40, 2002, pp. 97-110.
- BILLIONNET A., *Using integer programming to solve the train platforming problem*. Transportation Science, 37, 2003, pp. 213-222.
- BILLIONNET A., *Minimising total average cycle stock subject to practical constraints*. Journal of the Operational Research Society, 54, 2003, pp. 362-370.
- BILLIONNET A., *Mixed integer programming for the 0-1 maximum probability model*. European Journal of Operational Research, 156, 2004, pp. 83-91.

Index

A

ADN, 328
ajout de contraintes, 140
algorithme
 affine-scaling, 84, 86
 affine-scaling (interprétation géométrique), 89, 92
 branch and bound, 121
 de Dinkelbach, 47, 345
 de résolution des programmes quadratiques convexes à contraintes linéaires, 107
 de résolution des programmes quadratiques convexes à contraintes linéaires (interprétation géométrique), 110
 de troncatures, 130, 133
 de troncatures (interprétation géométrique), 136
 du simplexe, 14, 41, 62, 64, 83, 137
 du simplexe (finitude), 81
 du simplexe (initialisation), 76
 dual du simplexe, 96, 124, 133
 polynomial, 5, 9, 10, 11, 116
alphabet, 327
AMPL, 12, 13, 14, 16, 18, 19, 22, 36
approximation, 201, 399, 416
arborescence partielle, 284
arbre de recherche, 14, 19, 37, 121, 128
arbre partiel de poids minimal, 284

B

base, 57, 61
biknapsack, 116
biologie moléculaire, 328
bipartition d'un graphe, 251, 403
borne à la racine de l'arbre, 139

borne de la valeur optimale, 121, 339, 358, 370, 375, 378, 379, 380, 389, 396, 398, 414, 421, 423, 425, 432
borne de variables, 335, 337
branchement, 137, 140

C

carré d'une fonction linéaire, 215
carré d'une variable entière, 216
changement de variables, 164
chemin, 270
choix
 d'une formulation, 4, 20, 21, 143, 144, 147, 333
 des coefficients, 160, 162
 des contraintes, 173
 des variables, 164
choix d'investissements, 175, 253
circuit hamiltonien, 296
clique, 174, 350, 354, 385, 393
clique de cardinal maximal, 350, 353
coefficient de linéarisation, 161, 362, 363, 365, 368, 373
coloration, 349
comparaison de relaxations continues, 181, 184
complexité, 9, 11
conditions d'optimalité, 102
conditions de Karush, Kuhn et Tucker, 103
connaissance d'une bonne solution, 338, 339, 342, 345, 347, 348, 396
contrainte
 disjonctive, 163
 linéaire, 2, 3, 41
 quadratique, 3, 27
 redondante, 20, 180, 333, 334, 336, 379, 384, 387
convexification, 21, 25, 35, 37, 152, 204, 230, 400, 404
coupe, 131, 184, 379, 384