

Devoir maison n°7

À faire pour le lundi 12/01

Programmation linéaire et coloration de graphe

1 Programmation linéaire

La programmation linéaire (*Linear Programming* ou LP) est un problème d'optimisation qui peut se présenter sous différents formats selon les contraintes imposées. La forme la plus générale qu'on peut lui donner est la suivante : **Programmation Linéaire (PL)** :

- * **Instance** : une matrice $A \in \mathcal{M}_n(\mathbb{Q})$ et deux vecteurs $B, C \in \mathcal{M}_{n,1}(\mathbb{Q})$.
- * **Solution** : un vecteur $X \in \mathcal{M}_{n,1}(\mathbb{Q})$ vérifiant $AX \leq B$.
- * **Optimisation** : Maximiser $C^T X$.

Il a été montré en 1979 que PL peut être résolu en temps polynomial, mais nous ne nous intéresserons pas aux algorithmes permettant de résoudre ce problème, dont la complexité temporelle a été améliorée jusqu'à environ $\mathcal{O}(n^{2,1} \log M)$ où M est la plus grande valeur absolue d'un coefficient d'une des matrices.

Ce problème est important car un grand nombre de problèmes d'optimisation peuvent être formulés en termes de programmation linéaire, avec plus ou moins de contraintes.

1.1 Programmation entière

On considère deux variantes de décision du problème :

- **Programmation Entière (PE)** :
 - * **Instance** : une matrice $A \in \mathcal{M}_n(\mathbb{Q})$ et un vecteur $B \in \mathcal{M}_{n,1}(\mathbb{Q})$.
 - * **Question** : existe-t-il un vecteur $X \in \mathcal{M}_{n,1}(\mathbb{Z})$ vérifiant $AX \leq B$?
- **Équations Zéro-Un (EZU)** :
 - * **Instance** : une matrice $A \in \mathcal{M}_n(\mathbb{Q})$ et un vecteur $B \in \mathcal{M}_{n,1}(\mathbb{Q})$.
 - * **Question** : existe-t-il un vecteur $X \in \mathcal{M}_{n,1}(\{0,1\})$ vérifiant $AX \leq B$?

Le premier problème consiste à déterminer l'existence d'un vecteur à coefficients entiers (même si ceux des entrées ne le sont pas), le deuxième à coefficients dans $\{0,1\}$. Cette modification qui peut sembler anodine transforme un problème dans P en un problème NP-complet.

Question 1 Montrer que EZU \leq_m^P PE.

Question 2 Expliquer pourquoi il n'est pas aussi facile de montrer que PE \in NP que de montrer que EZU \in NP.

Pour la suite, on admet que PE \in NP. On pourra se référer au livre *Integer Programming* de Conforti, Cornuejols, et Zambelli pour une preuve en 35 lemmes, propositions et théorèmes de ce résultat.

On cherche à prouver la NP-complétude par une réduction depuis 3-SAT. On considère une formule en 3-FNC $\varphi = \bigwedge_{i=1}^m C_i$ où C_i est une clause disjonctive à trois littéraux, sur l'ensemble de variables $\mathcal{V} = \{v_1, \dots, v_n\}$.

On suppose que chaque C_i ne contient qu'au plus une occurrence de chaque variable (avec une négation ou non).

Question 3 Soit $\mu \in \{0, 1\}^V$ une valuation et $C = \ell_1 \vee \ell_2 \vee \ell_3$ une clause de φ . Montrer que $\mu(C) = 1$ si et seulement si $\mu(\ell_1) + \mu(\ell_2) + \mu(\ell_3) \geq 1$.

Question 4 En déduire 3-SAT \leq_p EZU.

Indication : on cherchera un vecteur X de la forme $X = (\mu(v_1)\mu(v_2)\cdots\mu(v_n))^T$.

Question 5 En déduire que EZU et PE sont NP-complets.

Pour toute la suite, on utilisera les noms PE et EZU pour désigner les variantes d'optimisation des deux problèmes précédents (c'est-à-dire dont la formulation est similaire à PL en imposant $X \in \mathcal{M}_{n,1}(\mathbb{Z})$ ou $X \in \mathcal{M}_{n,1}(\{0, 1\})$).

1.2 Couverture des arêtes par les sommets

On considère ici la variante d'optimisation du problème de couverture par les sommets déjà étudié dans les chapitres précédents : **Couverture par sommets (CPS)** :

- * **Instance** : un graphe non orienté $G = (S, A)$.
- * **Solution** : une couverture des arêtes par les sommets C , c'est-à-dire un ensemble $C \subseteq S$ tel que pour tout $a \in A$, $a \cap C \neq \emptyset$.
- * **Optimisation** : minimiser $|C|$.

On suppose pour simplifier que $S = \llbracket 1, n \rrbracket$ pour tout graphe $G = (S, A)$ considéré.

On considère une formulation de EZU comme :

- * **Instance** : un graphe non orienté $G = (S, A)$.
- * **Solution** : un vecteur $X \in \mathcal{M}_{n,1}(\{0, 1\})$ vérifiant $x_i + x_j \geq 1$ pour tout $\{i, j\} \in A$.
- * **Optimisation** : minimiser $\sum_{i=1}^n x_i$.

Question 6 Justifier que trouver la solution au problème précédent est équivalent à trouver une solution de CPS.

On considère maintenant une formulation de PL comme :

- * **Instance** : un graphe non orienté $G = (S, A)$.
- * **Solution** : un vecteur $X \in \mathcal{M}_{n,1}(\mathbb{Q})$ vérifiant $0 \leq x_i \leq 1$ pour tout $i \in S$ et $x_i + x_j \geq 1$ pour tout $\{i, j\} \in A$.
- * **Optimisation** : minimiser $\sum_{i=1}^n x_i$.

Cette généralisation du problème est appelée **relaxation continue** (on passe d'un problème discret « difficile » à un problème continu « facile »).

On note $OPT_{EZU}(G)$ et $OPT_{PL}(G)$ la valeur minimale $\sum_{i=1}^n x_i$ obtenue dans les deux problèmes précédents.

Question 7 Montrer que pour un graphe G , $OPT_{PL}(G) \leq OPT_{EZU}(G)$.

Si $X = (x_1 x_2 \cdots x_n)^T$ est une solution au problème PL précédent, on note $Y = (y_1 y_2 \cdots y_n)^T$ défini par $y_i = \begin{cases} 0 & \text{si } x_i < \frac{1}{2} \\ 1 & \text{sinon} \end{cases}$, c'est-à-dire la solution entière obtenue en arrondissant la solution continue.

Question 8 Montrer que $\sum_{i=1}^n y_i \leq 2OPT_{EZU}(G)$.

Question 9 En déduire une 2-approximation de CPS.

1.3 Couverture d'ensemble

On considère le problème **Couverture d'Ensemble** :

- * **Instance** : un entier N et une collections d'ensembles $S_1, \dots, S_n \in \mathcal{P}(\llbracket 1, N \rrbracket)$.
- * **Solution** : un sous-ensemble $I \subseteq \llbracket 1, n \rrbracket$ tel que $\bigcup_{i \in I} S_i = \llbracket 1, N \rrbracket$.
- * **Optimisation** : minimiser $|I|$.

Pour $x \in \llbracket 1, N \rrbracket$, on note $J(x) = \{j \in \llbracket 1, n \rrbracket \mid x \in S_j\}$.

Question 10 Donner une formulation équivalente de ce problème comme un problème EZU.

La relaxation continue de ce problème peut se formuler comme

- * **Instance** : un entier N et une collections d'ensembles $S_1, \dots, S_n \in \mathcal{P}(\llbracket 1, N \rrbracket)$.
- * **Solution** : un vecteur $X \in \mathcal{M}_{n,1}(\mathbb{R})$ vérifiant $0 \leq x_i \leq 1$ pour tout $i \in \llbracket 1, n \rrbracket$ et $\sum_{j \in J(x)} x_j \geq 1$ pour tout $x \in \llbracket 1, N \rrbracket$.
- * **Optimisation** : minimiser $\sum_{i=1}^n x_i$.

Si $X = (x_1 x_2 \cdots x_n)^T$ est une solution au problème PL précédent, on définit l'algorithme **Couv_Alea** suivant :

Début algorithme

```

 $I \leftarrow \emptyset.$ 
Pour  $i = 1$  à  $n$  Faire
     $\quad \lfloor$  Ajouter  $i$  à  $I$  avec probabilité  $x_i$ .
Renvoyer  $I$ .
```

Question 11 Montrer que si I est l'ensemble renvoyé par l'algorithme précédent, alors $\mathbb{E}(|I|)$ est inférieur ou égal à la taille d'une solution optimale à **Couverture d'Ensemble**.

Question 12 Montrer que pour $x \in \llbracket 1, N \rrbracket$, $\mathbb{P}\left(x \in \bigcup_{i \in I} S_i\right) \geq 1 - \frac{1}{e}$.

Indication : on pourra utiliser, après l'avoir montrée, l'inégalité $1 - t \leq e^{-t}$ pour $t \geq 0$.

Pour $k \in \mathbb{N}^*$, on définit maintenant l'algorithme **Couv_Alea2** :

Début algorithme

```

 $I \leftarrow \emptyset.$ 
Pour  $i = 1$  à  $\lceil 2 \ln N \rceil$  Faire
     $\quad \lfloor$   $I \leftarrow I \cup \text{Couv_Alea}()$ .
Renvoyer  $I$ .
```

Question 13 Montrer que si I est l'ensemble renvoyé par l'algorithme précédent, alors pour $x \in \llbracket 1, N \rrbracket$, $\mathbb{P}\left(x \in \bigcup_{i \in I} S_i\right) \geq 1 - \frac{1}{N^2}$.

Question 14 En déduire que l'algorithme précédent renvoie une couverture de l'ensemble $\llbracket 1, N \rrbracket$ avec une probabilité $\geq 1 - \frac{1}{N}$.

Question 15 On note I^* une solution optimale à Couverture d'Ensemble. Montrer que :

$$\mathbb{P}(|I| \leq (4 \ln N + 2)|I^*|) \geq \frac{1}{2}$$

Indication : on pourra utiliser l'inégalité de Markov.

Remarque

La méthode de relaxation continue est parfois utilisée comme heuristique d'évaluation dans un algorithme *Branch and Bound*. C'est cette méthode qui a été appliquée en considérant le problème Sac à dos fractionnaire.

2 Coloration de graphe

Pour un graphe $G = (S, A)$, une *numérotation* de G est une fonction $c : S \rightarrow \mathbb{N}$. Pour c une numérotation de G et $s \in S$, on appelle *couleur* de s la valeur $c(s)$. Une *coloration* de G est une numérotation de G telle que deux sommets adjacents n'ont jamais la même couleur. Pour un entier $k \in \mathbb{N}^*$, une k -coloration de G est une coloration de G à valeurs dans $\llbracket 0, k - 1 \rrbracket$, c'est-à-dire une fonction $c : S \rightarrow \{0, \dots, k - 1\}$ telle que pour toute arête $\{s, t\} \in A$, $c(s) \neq c(t)$.

La figure 1 donne deux numérotations possibles d'un même graphe G_0 . La numérotation de gauche n'est pas une coloration car les sommets grisés (1 et 4) sont adjacents et ont la même couleur. La numérotation de droite est une 4-coloration. Attention, dans les représentations de la figure 1, les nombres à l'extérieur des sommets ne correspondent pas aux indices des sommets mais aux couleurs. Les indices des sommets sont à l'intérieur.

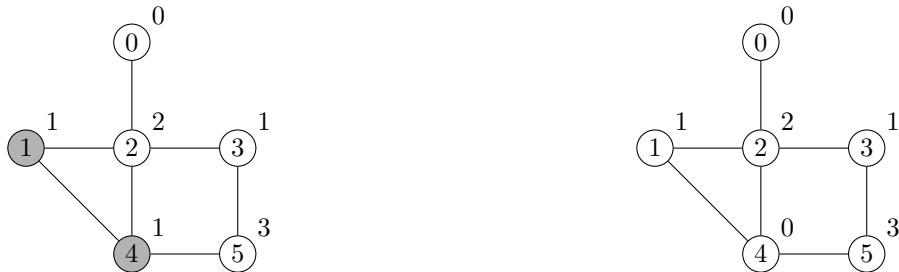


FIGURE 1 – Deux numérotations de G_0 .

G est dit k -colorable s'il possède une k -coloration. On définit également le *nombre chromatique* de G , noté $\chi(G)$, comme étant le plus petit entier k tel que G est k -colorable. Par exemple, le nombre chromatique du graphe G_0 est 3.

Question 16 Déterminer le nombre chromatique du graphe G_1 représenté figure 2. On exhibera une coloration optimale, et on justifiera rigoureusement qu'il est impossible d'utiliser moins de numéros différents.

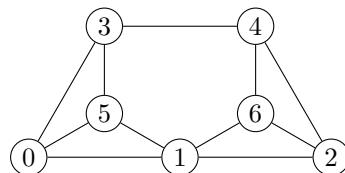


FIGURE 2 – Le graphe G_1 .

On cherche à montrer que le problème général du calcul du nombre chromatique d'un graphe est difficile.

Pour ce faire, on étudie plutôt des problèmes de décision liés à la coloration. On définit les problèmes de décision suivants :

- Problème **k-coloration** (pour k fixé)
 - * Entrée : un graphe G .
 - * Question : le graphe G est-il k -colorable ?
- Problème **Coloration**
 - * Entrée : un graphe G et un entier k .
 - * Question : le graphe G est-il k -colorable ?

Par ailleurs, pour A et B deux problèmes de décision, la notation $A \leq_m^p B$ signifie qu'il existe une réduction many-one polynomiale de A à B .

Question 17 Montrer que pour tout entier k , **k-coloration** \leq_m^p **Coloration**.

Question 18 Montrer que pour $h \leq k$ deux entiers fixés, **h-coloration** \leq_m^p **k-coloration**.

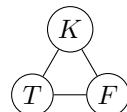
Question 19 Montrer que **2-coloration** $\in P$. On décrira en français ou pseudo-code un algorithme de complexité linéaire permettant de résoudre le problème.

Les deux questions précédentes montrent que les problèmes **1-coloration** et **2-coloration** sont « faciles ». Les questions suivantes montrent la **NP-complétude** de **3-coloration**.

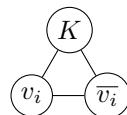
On admet que le problème **3-SAT** est **NP-complet**. La suite de cette partie a pour objectif de montrer qu'il se réduit en temps polynomial au problème **3-coloration**. Pour ce faire, en considérant une formule booléenne φ en 3-FNC, on construit un graphe G_φ qui est 3-colorable si et seulement si φ est satisfiable. L'idée est que deux des couleurs utilisées correspondent à l'affectation *Vrai* ou *Faux* pour chaque littéral, et la troisième couleur est une couleur de contrôle.

Dans le détail, si φ est une formule en 3-FNC sur un ensemble $V = \{v_1, \dots, v_n\}$ de n variables, alors on définit le graphe $G_\varphi = (S, A)$ par :

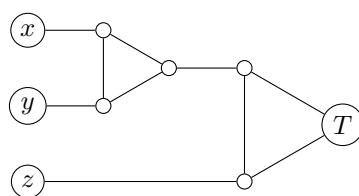
- S contient trois sommets T (*True*), F (*False*) et K (*Control*), un sommet par littéral possible, et 5 sommets supplémentaires par clause ;
- A contient les arêtes suivantes :
 - * $\{T, F\}, \{T, K\}, \{F, K\}$



* pour chaque variable v_i , les arêtes $\{v_i, \bar{v}_i\}, \{v_i, K\}, \{\bar{v}_i, K\}$



* pour chaque clause $C = x \vee y \vee z$ apparaissant dans φ (où x, y, z sont des littéraux), on rajoute les 10 arêtes telles que décrites dans le graphe ci-dessous, en utilisant les 5 sommets supplémentaires associés à la clause.



Par exemple, la figure 3 représente le graphe G_{φ_0} obtenu pour la formule booléenne φ_0 sur l'ensemble de variables $\{a, b, c, d\}$.

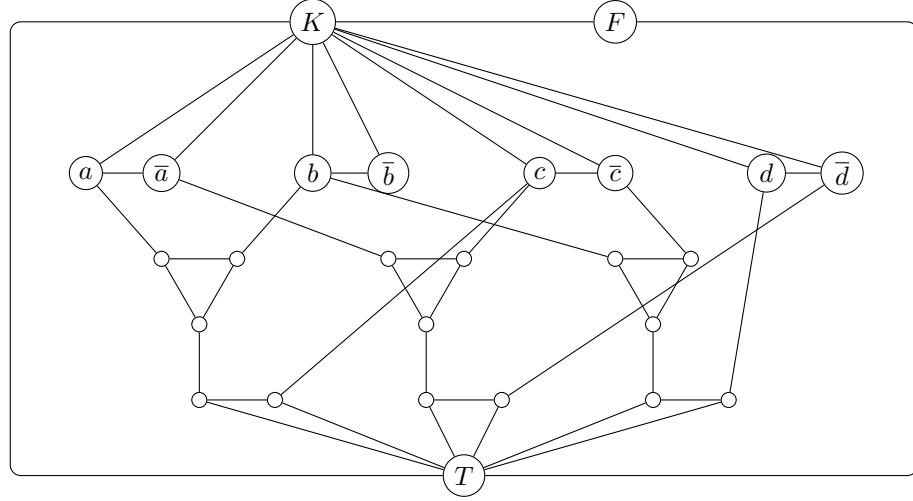


FIGURE 3 – Le graphe G_{φ_0} pour $\varphi_0 = (a \vee b \vee c) \wedge (\bar{a} \vee c \vee \bar{d}) \wedge (b \vee \bar{c} \vee d)$.

Question 20 Justifier que la construction de G_φ à partir d'une formule φ en 3-FNC est en temps polynomial en fonction de la taille de φ (la taille d'une formule en 3-FNC est son nombre de clauses).

Question 21 Montrer que le graphe de la figure 4 est 3-colorable et que pour toute 3-coloration, au moins l'un des sommets x , y ou z a la même couleur que T .

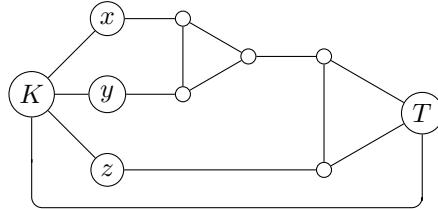


FIGURE 4 – Un gadget de clause.

Question 22 On suppose que φ est satisfiable. Montrer que G_φ est 3-colorable. On expliquera comment construire une 3-coloration de G_φ à partir d'une valuation de φ .

Question 23 On suppose que G_φ est 3-colorable. Montrer que φ est satisfiable. On expliquera comment construire une valuation de φ à partir d'une coloration de G_φ .

Question 24 En déduire que 3-coloration et Coloration sont NP-complets.
