

M1 Info - Optimisation et Recherche Opérationnelle

Cours 6 - Programmation lineaire et en nombres entiers

Flot maximum, coupe minimum et couverture par sommets

Semestre Automne 2020-2021 - Université Claude Bernard Lyon 1

Christophe Crespelle
christophe.crespelle@inria.fr



département

Informatique

Faculté des Sciences et Technologies
Université Claude Bernard Lyon 1

Programmation lineaire

Probleme en forme standard :

- **Entrée :**

- ▶ m reels b_1, b_2, \dots, b_m
- ▶ n reels c_1, c_2, \dots, c_n
- ▶ mn reels a_{ij} pour $i \in \llbracket 1, m \rrbracket$ et $j \in \llbracket 1, n \rrbracket$

- **Sortie :** n nombres reels x_1, x_2, \dots, x_n qui

- ▶ maximisent $\sum_{j=1}^n c_j x_j$
- ▶ sous les contraintes :
 - ▶ $\sum_{j=1}^n a_{ij} x_j \leq b_i$, pour $i \in \llbracket 1, m \rrbracket$ et
 - ▶ $x_j \geq 0$, pour tout $j \in \llbracket 1, n \rrbracket$

Difficulte de calcul : **Polynomial**

Mise en forme matricielle ^{*n variables*}

maximiser $c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4$ sous contraintes :

$$\left(\begin{array}{cccc|c} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & a_{14}x_4 & \leq & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & a_{24}x_4 & \leq & b_2 \\ a_{31}x_1 & + & a_{32}x_2 & + & a_{33}x_3 & + & a_{34}x_4 & \leq & b_3 \end{array} \right) \left| \begin{array}{l} \\ \\ \end{array} \right. \text{m inéquations}$$

$$\forall i \in \llbracket 1, 4 \rrbracket, x_i \geq 0$$

Mise en forme matricielle

maximiser $c_1x_1 + c_2x_2 + c_3x_3 + c_4x_4$ sous contraintes :

$$a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 \leq b_1$$

$$a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 \leq b_2$$

$$a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 \leq b_3$$

$$\forall i \in \llbracket 1, 4 \rrbracket, x_i \geq 0$$

maximiser :

$$\begin{bmatrix} c_1 & c_2 & c_3 & c_4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

sous contraintes :

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$\text{et } \forall i \in \llbracket 1, 4 \rrbracket, x_i \geq 0$$

Rappel produit matriciel

$$\begin{bmatrix} c_1 & c_2 & c_3 & c_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = [c_1 x_1 + c_2 x_2 + c_3 x_3 + c_4 x_4]$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} a_{11} x_1 + a_{12} x_2 + a_{13} x_3 + a_{14} x_4 \\ a_{21} x_1 + a_{22} x_2 + a_{23} x_3 + a_{24} x_4 \\ \dots \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

Programmation lineaire : exemple

Probleme : Un magasin veut faire fabriquer des pantalons et des vestes de sport a une usine textile qui dispose de $750m^2$ de coton et $1000m^2$ de polyester. La fabrication d'une paire de pantalon requiert $1m^2$ de coton et $2m^2$ de polyester et celle d'une veste requiert $1.5m^2$ de coton et $1m^2$ de polyester. Sachant que le benefice du magasin sur une paire de pantalon est de 50\$ et de 40\$ pour une veste, quelle est la commande que doit passer le magasin pour maximiser son benefice total ?

Programmation lineaire : exemple

Probleme : Un magasin veut faire fabriquer des pantalons et des vestes de sport a une usine textile qui dispose de $750m^2$ de coton et $1000m^2$ de polyester. La fabrication d'une paire de pantalon requiert $1m^2$ de coton et $2m^2$ de polyester et celle d'une veste requiert $1.5m^2$ de coton et $1m^2$ de polyester. Sachant que le benefice du magasin sur une paire de pantalon est de 50\$ et de 40\$ pour une veste, quelle est la commande que doit passer le magasin pour maximiser son benefice total ?

Modelisation : x le nombre de pantalons commandes et y celui de vestes. On veut :

- maximiser $50x + 40y$
- sous les contraintes :

Programmation lineaire : exemple

Probleme : Un magasin veut faire fabriquer des pantalons et des vestes de sport a une usine textile qui dispose de $750m^2$ de coton et $1000m^2$ de polyester. La fabrication d'une paire de pantalon requiert $1m^2$ de coton et $2m^2$ de polyester et celle d'une veste requiert $1.5m^2$ de coton et $1m^2$ de polyester. Sachant que le benefice du magasin sur une paire de pantalon est de 50\$ et de 40\$ pour une veste, quelle est la commande que doit passer le magasin pour maximiser son benefice total ?

Modelisation : x le nombre de pantalons commandes et y celui de vestes. On veut :

- maximiser $50x + 40y$
- sous les contraintes :
 - ▶ $x + 1.5y \leq 750$

$$2x + y \leq 1000$$

Programmation lineaire : exemple

Probleme : Un magasin veut faire fabriquer des pantalons et des vestes de sport a une usine textile qui dispose de $750m^2$ de coton et $1000m^2$ de polyester. La fabrication d'une paire de pantalon requiert $1m^2$ de coton et $2m^2$ de polyester et celle d'une veste requiert $1.5m^2$ de coton et $1m^2$ de polyester. Sachant que le benefice du magasin sur une paire de pantalon est de 50\$ et de 40\$ pour une veste, quelle est la commande que doit passer le magasin pour maximiser son benefice total ?

Modelisation : x le nombre de pantalons commandes et y celui de vestes. On veut :

- maximiser $50x + 40y$
- sous les contraintes :
 - ▶ $x + 1.5y \leq 750$
 - ▶ $2x + y \leq 1000$

Programmation lineaire : exemple

Probleme : Un magasin veut faire fabriquer des pantalons et des vestes de sport a une usine textile qui dispose de $750m^2$ de coton et $1000m^2$ de polyester. La fabrication d'une paire de pantalon requiert $1m^2$ de coton et $2m^2$ de polyester et celle d'une veste requiert $1.5m^2$ de coton et $1m^2$ de polyester. Sachant que le benefice du magasin sur une paire de pantalon est de 50\$ et de 40\$ pour une veste, quelle est la commande que doit passer le magasin pour maximiser son benefice total ?

Modelisation : x le nombre de pantalons commandes et y celui de vestes. On veut :

- maximiser $50x + 40y$

- sous les contraintes :

▶ $x + 1.5y \leq 750$

▶ $2x + y \leq 1000$

▶ $x \geq 0$ et $y \geq 0$

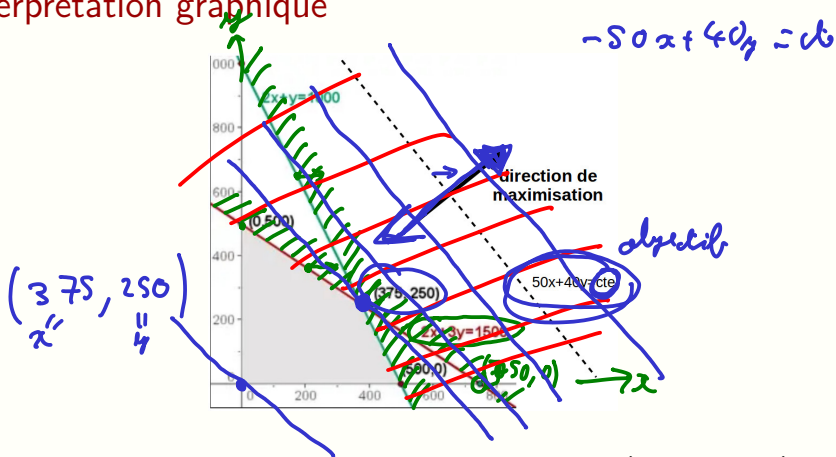
$$50x + 40y = \underline{\text{de}}$$

$$2x + 3y = 1500$$

$$x + 1.5y = 750$$

$$\rightarrow 2x + y = 1000$$

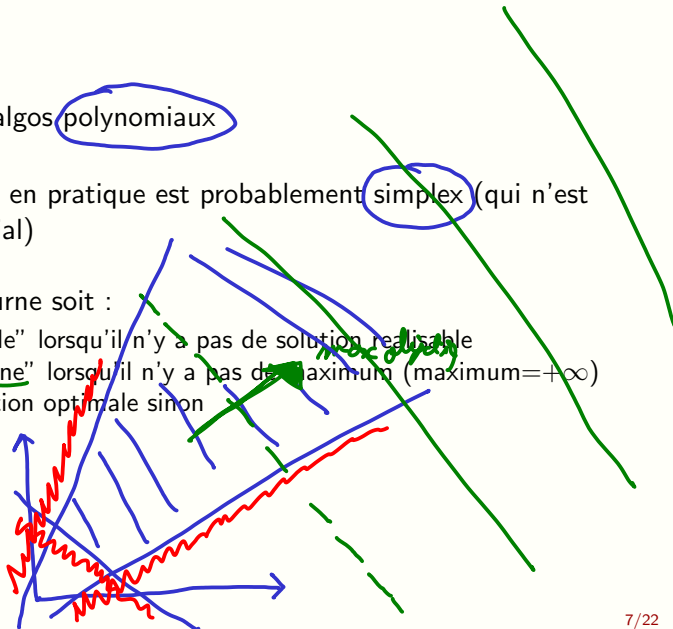
Interpretation graphique



- chaque contrainte correspond a un demi-plan (demi-espace) delimité par une droite (hyperplan)
- la region des solutions faisables est un polyedre
- une solution maximum (lorsqu'elle existe) est atteinte sur un des sommets du polyedre

Resolution de programmes lineaires

- il existe des algos polynomiaux
- le plus utilise en pratique est probablement simplex (qui n'est pas polynomial)
- simplex retourne soit :
 - ▶ "infaisable" lorsqu'il n'y a pas de solution réalisable
 - ▶ "non borne" lorsqu'il n'y a pas de maximum (maximum = $+\infty$)
 - ▶ une solution optimale sinon



Formes non standard

- la fonction objectif peut être une minimisation plutôt qu'une maximisation
 - ▶ on multiplie par -1 : $\min \rightarrow \max$
- il peut y avoir des contraintes de négativité sur les variables
 - ▶ $x_i \rightarrow -x_i$
- il peut y avoir des variables sans contraintes de positivité
 - ▶ $x_i \rightarrow x_i^+ - x_i^-$ avec $x_i^+ \geq 0$ et $x_i^- \geq 0$
- il peut y avoir des contraintes d'inégalité qui soient \geq à la place de \leq
 - ▶ on multiplie par -1 : $\geq \rightarrow \leq$
- il peut y avoir des contraintes d'égalité plutôt que d'inégalité
 - ▶ $= \rightarrow \leq$ et \geq , ou bien
 - ▶ on élimine une variable

maximize
minimize

$$\begin{array}{r} -4x - 2y - 3z \\ 4x + 2y + 3z \end{array}$$

$$\begin{array}{r} -3x' - 2y' \leq -25 \\ -3y^+ + 3y^- = 5 \\ 4x' - 2y' = 5 \\ 2x' + 5y' \leq 3 \\ x' \leq 0 \\ x' \geq 0 \\ y' \geq 0 \end{array}$$

$y = y^+ - y^-$

$x \leq 4$

pas \leq ou $>$

avec $y^+ \geq 0$ et $y^- \geq 0$

$$\begin{array}{r} -4x' - 3y^+ + 3y^- \leq 5 \\ - \quad - \quad - \quad \neq -5 \end{array}$$

Primal / dual

Primal :

maximiser $\sum_{j=1}^n c_j x_j$ sous contraintes :

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \text{ pour } i \in \llbracket 1, m \rrbracket$$

$$x_j \geq 0, \text{ pour } j \in \llbracket 1, n \rrbracket$$

Primal / dual

Primal :

maximiser $\sum_{j=1}^n c_j x_j$ sous contraintes :

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \text{ pour } i \in \llbracket 1, m \rrbracket$$

$$x_j \geq 0, \text{ pour } j \in \llbracket 1, n \rrbracket$$

Dual :

minimiser $\sum_{i=1}^m b_i y_i$ sous contraintes :

$$\sum_{i=1}^m a_{ij} y_i \geq c_j, \text{ pour } j \in \llbracket 1, n \rrbracket$$

$$y_i \geq 0, \text{ pour } i \in \llbracket 1, m \rrbracket$$

Primal / dual

Primal : $f(x)$

maximiser $\sum_{j=1}^n c_j x_j$ sous contraintes :

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \text{ pour } i \in \llbracket 1, m \rrbracket$$

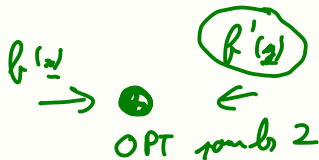
$$x_j \geq 0, \text{ pour } j \in \llbracket 1, n \rrbracket$$

Dual : $f'(y)$

minimiser $\sum_{i=1}^m b_i y_i$ sous contraintes :

$$\sum_{i=1}^m a_{ij} y_i \geq c_j, \text{ pour } j \in \llbracket 1, n \rrbracket$$

$$y_i \geq 0, \text{ pour } i \in \llbracket 1, m \rrbracket$$



x une sol. réalisable
du primal

y une sol. réalisable
du dual

$$f(x) \leq f'(y)$$

= OPT

Théorème (Dualité forte)

Si le primal admet une solution optimale, alors le dual aussi et les valeurs objectives optimales des deux problèmes sont égales.

Primal / dual

Primal :

50 \$
40 \$
4 var.

maximiser $c_1 x_1 + c_2 x_2 + c_3 x_3 + c_4 x_4$ sous contraintes :

$$\begin{array}{rcccccc} a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & a_{14}x_4 & \leq & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_3 & + & a_{24}x_4 & \leq & b_2 \\ a_{31}x_1 & + & a_{32}x_2 & + & a_{33}x_3 & + & a_{34}x_4 & \leq & b_3 \end{array}$$

$$\forall i \in \llbracket 1, 4 \rrbracket, x_i \geq 0$$

3 eqm.
4₁
4₂
4₃

Dual :

3 var.

minimiser $b_1 y_1 + b_2 y_2 + b_3 y_3$ sous contraintes :

$$\begin{array}{rcccccc} a_{11}y_1 & + & a_{21}y_2 & + & a_{31}y_3 & \geq & c_1 \\ a_{12}y_1 & + & a_{22}y_2 & + & a_{32}y_3 & \geq & c_2 \\ a_{13}y_1 & + & a_{23}y_2 & + & a_{33}y_3 & \geq & c_3 \\ a_{14}y_1 & + & a_{24}y_2 & + & a_{34}y_3 & \geq & c_4 \end{array}$$

$$\forall i \in \llbracket 1, 3 \rrbracket, y_i \geq 0$$

4 eq.
x₁
x₂
x₃
x₄

Primal / dual

Primal :

maximiser $\begin{bmatrix} c_1 & c_2 & c_3 & c_4 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$

sous contraintes $\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$

Dual :

minimiser $\begin{bmatrix} b_1 & b_2 & b_3 \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix}$

sous contraintes $\begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \\ a_{13} & a_{23} & a_{33} \\ a_{14} & a_{24} & a_{34} \end{bmatrix} \cdot \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \geq \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix}$

Retour sur l'algorithme du simplexe

- il existe des algos polynomiaux
- le plus utilise en pratique est probablement simplex (qui n'est pas polynomial)
- simplex retourne soit :
 - ▶ "infaisable" lorsqu'il n'y a pas de solution realisable
 - ▶ "non borne" lorsqu'il n'y a pas de maximum (maximum= $+\infty$)
 - ▶ une solution optimale, sinon
- il existe une solution optimale pour le primal ssi il existe une solution optimale pour le dual et
- les valeurs objectifs de ces solutions optimales sont les memes pour le primal et le dual.

Retour sur l'algorithme du simplexe

- il existe des algos polynomiaux
- le plus utilise en pratique est probablement simplex (qui n'est pas polynomial)
- simplex retourne soit :
 - ▶ "infaisable" lorsqu'il n'y a pas de solution realisable
 - ▶ "non borne" lorsqu'il n'y a pas de maximum (maximum= $+\infty$)
 - ▶ une solution optimale, sinon
- il existe une solution optimale pour le primal ssi il existe une solution optimale pour le dual et
- les valeurs objectifs de ces solutions optimales sont les memes pour le primal et le dual.
- De plus, dans ce cas, simplex fournit aussi une solution optimale pour le dual.

Un exemple plus avancé : flot maximum

Q: que sont les variables ???
 fl (u,v) $\forall (u,v) \in A$ et f

Cost de la fonction obj. ??

$\textcircled{1} f + \textcircled{2} f(u,v) + \textcircled{3} f(u',v') \dots$

$|A| + 1$

variable 0

non 1

(la cost de f)

capacité maximiser f sous contraintes :

$f(u, v) \leq c(u, v), \forall (u, v) \in A$

def " $\sum_{v \in N^+(s)} f(s, v) - \sum_{v \in N^-(s)} f(v, s) = f$

def de $\sum_{v \in N^+(t)} f(t, v) - \sum_{v \in N^-(t)} f(v, t) = -f$

conservation $\sum_{v \in N^+(u)} f(u, v) = \sum_{v \in N^-(u)} f(v, u), \forall u \in V \setminus \{s, t\}$

$f(u, v) \geq 0, \forall (u, v) \in A$

Passage au dual : forme standard du primal

Primal sous forme standard :

maximiser f sous contraintes :

$$f(u, v) \leq c(u, v), \forall (u, v) \in A$$

$$\sum_{v \in N^+(s)} f(s, v) - \sum_{v \in N^-(s)} f(v, s) - f \leq 0$$

$$- \sum_{v \in N^+(s)} f(s, v) + \sum_{v \in N^-(s)} f(v, s) + f \leq 0$$

$$\sum_{v \in N^+(t)} f(t, v) - \sum_{v \in N^-(t)} f(v, t) + f \leq 0$$

$$- \sum_{v \in N^+(t)} f(t, v) + \sum_{v \in N^-(t)} f(v, t) - f \leq 0$$

$$\sum_{v \in N^+(u)} f(u, v) - \sum_{v \in N^-(u)} f(v, u) \leq 0, \forall u \in V \setminus \{s, t\}$$

$$- \sum_{v \in N^+(u)} f(u, v) + \sum_{v \in N^-(u)} f(v, u) \leq 0, \forall u \in V \setminus \{s, t\}$$

$$f(u, v) \geq 0, \forall (u, v) \in A \text{ et } f \geq 0$$

Passage au dual : forme standard du primal

Primal sous forme standard :

maximiser f sous contraintes :

$$f(u, v) \leq c(u, v), \forall (u, v) \in A \quad \rightarrow y_{uv}$$

$$\sum_{v \in N^+(s)} f(s, v) - \sum_{v \in N^-(s)} f(v, s) \leq 0 \quad \rightarrow y_s^+$$

$$- \sum_{v \in N^+(s)} f(s, v) + \sum_{v \in N^-(s)} f(v, s) \leq 0 \quad \rightarrow y_s^-$$

$$\sum_{v \in N^+(t)} f(t, v) - \sum_{v \in N^-(t)} f(v, t) \leq 0 \quad \rightarrow y_t^+$$

$$- \sum_{v \in N^+(t)} f(t, v) + \sum_{v \in N^-(t)} f(v, t) \leq 0 \quad \rightarrow y_t^-$$

$$\sum_{v \in N^+(u)} f(u, v) - \sum_{v \in N^-(u)} f(v, u) \leq 0, \forall u \in V \setminus \{s, t\} \quad \rightarrow y_u^+$$

$$- \sum_{v \in N^+(u)} f(u, v) + \sum_{v \in N^-(u)} f(v, u) \leq 0, \forall u \in V \setminus \{s, t\} \quad \rightarrow y_u^-$$

$$f(u, v) \geq 0, \forall (u, v) \in A \text{ et } f \geq 0$$

$$-y_s^+ + y_s^- + y_t^+ - y_t^- \geq 1$$

var. dual

$$y_{uv} + y_u^+ - y_u^- - y_v^+ + y_v^- \geq 0$$

Le dual

Dual :

minimiser $\sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

f
 $Q(u,v)$

$$-y_s^+ + y_s^- + y_t^+ - y_t^- \geq 1$$
$$y_{uv} + y_u^+ - y_u^- - y_v^+ + y_v^- \geq 0, \forall (u,v) \in A$$
$$y_{uv} \geq 0, \forall (u,v) \in A$$
$$y_u^+ \geq 0 \text{ et } y_u^- \geq 0, \forall (u,v) \in A$$

Le dual

Dual :

minimiser $\sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$-y_s^+ + y_s^- + y_t^+ - y_t^- \geq 1$$

$$y_{uv} + y_u^+ - y_u^- - y_v^+ + y_v^- \geq 0, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

$$y_u^+ \geq 0 \text{ et } y_u^- \geq 0, \forall (u,v) \in A \quad \rightarrow y_u$$

En posant $y_u^+ - y_u^- = y_u, \forall u \in V$, on obtient :

minimiser $\sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_{uv} + y_u - y_v \geq 0, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

pas de contrainte sur y_u

Le dual... c'est Coupe Minimum !

Dual :

minimiser $obj = \sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

Le dual... c'est Coupe Minimum !

Dual :

minimiser $obj = \sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

Démonstration.

- obj est minimum pour $y_{uv} = \max\{y_v - y_u, 0\}$

Le dual... c'est Coupe Minimum !

Dual :

minimiser $obj = \sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

Démonstration.

- obj est minimum pour $y_{uv} = \max\{y_v - y_u, 0\}$
- obj est invariant par translation : on peut prendre $y_s = 0$



Le dual... c'est Coupe Minimum !

Dual :

$$\text{minimiser } obj = \sum_{(u,v) \in A} c(u,v) y_{uv} \text{ sous contraintes :}$$

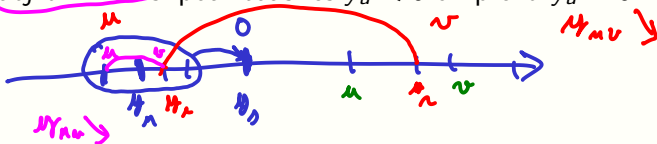
$$y_t - y_s \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A \quad \forall u, y_u \geq 0$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

Démonstration.

- obj est minimum pour $y_{uv} = \max\{y_v - y_u, 0\}$
- obj est invariant par translation : on peut prendre $y_s = 0$
- obj diminue si pour tous les $y_u < 0$ on prend $y_u = 0$



Le dual... c'est Coupe Minimum !

Dual :

minimiser $obj = \sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_u \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A$$

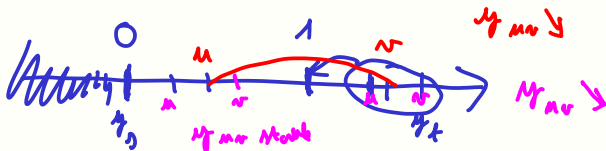
$$y_m \geq 0$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

$$y_m \leq 1$$

Démonstration.

- obj est minimum pour $y_{uv} = \max\{y_v - y_u, 0\}$
- obj est invariant par translation : on peut prendre $y_s = 0$
- obj diminue si pour tous les $y_u < 0$ on prend $y_u = 0$
- obj diminue si pour tous les $y_v > 1$ on prend $y_v = 1$



Le dual... c'est Coupe Minimum !

Dual :

minimiser $obj = \sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

Démonstration.

- obj est minimum pour $y_{uv} = \max\{y_v - y_u, 0\}$
- obj est invariant par translation : on peut prendre $y_s = 0$
- obj diminue si pour tous les $y_u < 0$ on prend $y_u = 0$
- obj diminue si pour tous les $y_v > 1$ on prend $y_v = 1$
($\Rightarrow y_t = 1$)

Le dual... c'est Coupe Minimum !

Dual :

minimiser $obj = \sum_{(u,v) \in A} c(u,v)y_{uv}$ sous contraintes :

$$y_t - y_s \geq 1$$

$$y_{uv} \geq y_v - y_u, \forall (u,v) \in A$$

$$y_{uv} \geq 0, \forall (u,v) \in A$$

Démonstration.

- obj est minimum pour $y_{uv} = \max\{y_v - y_u, 0\}$
- obj est invariant par translation : on peut prendre $y_s = 0$
- obj diminue si pour tous les $y_u < 0$ on prend $y_u = 0$
- obj diminue si pour tous les $y_v > 1$ on prend $y_v = 1$
($\Rightarrow y_t = 1$)
- considérons les bipartitions (U_c, V_c) définies pour $0 < c < 1$ par $U_c = \{u \in V \mid y_u < c\}$ et $V_c = \{v \in V \mid y_v > c\}$:
 $\exists c$ tel que obj diminue si $\forall u \in U_c$ on prend $y_u = 0$ et $\forall v \in V_c$ on prend $y_v = 1$

minimiser obj =

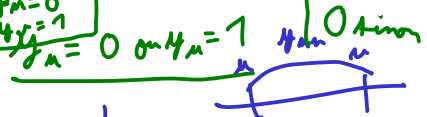
$\sum_{m,v} c(m,v) y_{mv}$

$y_a = 0$ et $y_A = 1$

~~$0 \leq y_m \leq 1$~~

$\sum_{m,v} c_{m,v}$
 $y_m = 0$
 $y_m = 1$

$\frac{c_{m,v}}{y_{m,v}} = 1$ si $y_m = 0$ et $y_v = 1$



$y_{mv} = \max \{ y_v - y_m, 0 \}$

S



$\sum_{m,v} C(m,v) y_{mv} = \sum_{\text{secteur}}$

$\sum_{\substack{m,v \\ m \leq l_1 \text{ et } v \geq r_1}} C(m,v) \times |r_1 - l_1|$

Programmation lineaire en nombres entiers

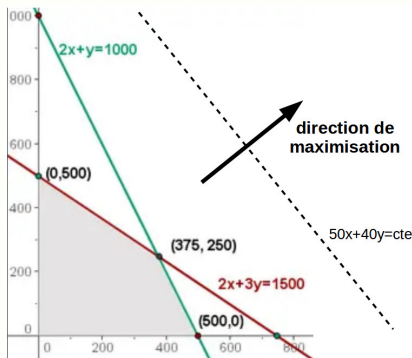
Définition

- un programme lineaire avec coefficients reels, comme precedemment
- mais on cherche les solutions entieres, c.a.d. avec tous les $x_i, i \in \llbracket 1, n \rrbracket$ entiers (contrainte d'integralite)

Programmation lineaire en nombres entiers

Définition

- un programme lineaire avec coefficients reels, comme precedemment
- mais on cherche les solutions entieres, c.a.d. avec tous les $x_i, i \in \llbracket 1, n \rrbracket$ entiers (contrainte d'integralite)



Programmation lineaire en nombres entiers

Définition

- un programme lineaire avec coefficients reels, comme precedemment
- mais on cherche les solutions entieres, c.a.d. avec tous les $x_i, i \in \llbracket 1, n \rrbracket$ entiers (contrainte d'integralite)

Complexite de calcul : NP-complet
(contrairement a la programmation lineaire)

Programmation lineaire en nombres entiers

Définition

- un programme lineaire avec coefficients reels, comme precedemment
- mais on cherche les solutions entieres, c.a.d. avec tous les $x_i, i \in \llbracket 1, n \rrbracket$ entiers (contrainte d'integralite)

Complexite de calcul : NP-complet
(contrairement a la programmation lineaire)

Resolution exacte : techniques de branching

Minimum Vertex cover (MVC)

- **Entrée** : un graphe $G = (V, E)$
(non orienté, simple et sans boucles)
- **Sortie** : un sous ensemble $S \subseteq V$ de sommets qui couvrent toutes les arêtes, c.a.d. tel que $\forall uv \in E, u \in S$ ou $v \in S$ et qui est de cardinalité minimum pour cette propriété.

Minimum Vertex cover (MVC)

- **Entrée** : un graphe $G = (V, E)$
(non orienté, simple et sans boucles)
- **Sortie** : un sous ensemble $S \subseteq V$ de sommets qui couvrent toutes les arêtes, c.a.d. tel que $\forall uv \in E, u \in S$ ou $v \in S$ et qui est de cardinalité minimum pour cette propriété.

Remarque

S est un vertex cover ssi $V \setminus S$ est un stable.

S est un minimum vertex cover ssi $V \setminus S$ est un stable maximum.

Minimum Vertex cover (MVC)

- **Entrée** : un graphe $G = (V, E)$
(non oriente, simple et sans boucles)
- **Sortie** : un sous ensemble $S \subseteq V$ de sommets qui couvrent toutes les aretes, c.a.d. tel que $\forall uv \in E, u \in S$ ou $v \in S$ et qui est de cardinalite minimum pour cette propriete.

Remarque

S est un vertex cover ssi $V \setminus S$ est un stable.

S est un minimum vertex cover ssi $V \setminus S$ est un stable maximum.

Conclusion : Stable Maximum et Minimum Vertex Cover sont en *quelque sorte* "equivalents" ... sauf que pas completement !

Minimum Vertex cover (MVC)

- **Entrée** : un graphe $G = (V, E)$
(non oriente, simple et sans boucles)
- **Sortie** : un sous ensemble $S \subseteq V$ de sommets qui couvrent toutes les aretes, c.a.d. tel que $\forall uv \in E, u \in S$ ou $v \in S$ et qui est de cardinalite minimum pour cette propriete.

Remarque

S est un vertex cover ssi $V \setminus S$ est un stable.

S est un minimum vertex cover ssi $V \setminus S$ est un stable maximum.

Conclusion : Stable Maximum et Minimum Vertex Cover sont en *quelque sorte* "equivalents" ... sauf que pas completement !

Complexite de calcul : Minimum Vertex Cover est NP-complet (car Stable Maximum l'est)

Minimum Vertex cover (MVC)

- **Entrée** : un graphe $G = (V, E)$
(non oriente, simple et sans boucles)
- **Sortie** : un sous ensemble $S \subseteq V$ de sommets qui couvrent toutes les aretes, c.a.d. tel que $\forall uv \in E, u \in S$ ou $v \in S$ et qui est de cardinalite minimum pour cette propriete.

Remarque

S est un vertex cover ssi $V \setminus S$ est un stable.

S est un minimum vertex cover ssi $V \setminus S$ est un stable maximum.

Conclusion : Stable Maximum et Minimum Vertex Cover sont en quelque sorte "equivalents" ... sauf que pas completement !

Complexite de calcul : Minimum Vertex Cover est NP-complet (car Stable Maximum l'est)

Approximabilite : Minimum Vertex Cover : OUI. Stable Maximum : NON !

Un prog. lineaire en nombres entiers pour Weighted MVC

Weighted : les sommets ont un poids w_i et on veut trouver S un MVC de poids $w(S)$ minimum (avec $w(S) = \sum_{i \in S} w_i$)

Pb plus general, donc au moins aussi dur

Un prog. lineaire en nombres entiers pour Weighted MVC

Weighted : les sommets ont un poids w_i et on veut trouver S un MVC de poids $w(S)$ minimum (avec $w(S) = \sum_{i \in S} w_i$)

Pb plus general, donc au moins aussi dur

En nombres entiers :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$x_i \in \{0, 1\}, \forall i \in V$$

Un prog. lineaire en nombres entiers pour Weighted MVC

Weighted : les sommets ont un poids w_i et on veut trouver S un MVC de poids $w(S)$ minimum (avec $w(S) = \sum_{i \in S} w_i$)

Pb plus general, donc au moins aussi dur

En nombres entiers :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$x_i \in \{0, 1\}, \forall i \in V$$

Relaxation lineaire :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$0 \leq x_i \leq 1, \forall i \in V$$

Un prog. lineaire en nombres entiers pour Weighted MVC

Weighted : les sommets ont un poids w_i et on veut trouver S un MVC de poids $w(S)$ minimum (avec $w(S) = \sum_{i \in S} w_i$)

Pb plus general, donc au moins aussi dur

En nombres entiers :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$x_i \in \{0, 1\}, \forall i \in V$$

Relaxation lineaire :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$0 \leq x_i \leq 1, \forall i \in V$$

Avantage : relaxation lineaire soluble en temps polynomial donne une solution reelle avec un meilleur objectif... mais non admissible.

Un prog. lineaire en nombres entiers pour Weighted MVC

Weighted : les sommets ont un poids w_i et on veut trouver S un MVC de poids $w(S)$ minimum (avec $w(S) = \sum_{i \in S} w_i$)

Pb plus general, donc au moins aussi dur

En nombres entiers :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$x_i \in \{0, 1\}, \forall i \in V$$

Relaxation lineaire :

minimiser $\sum_{i \in V} w_i x_i$ sous contraintes

$$x_i + x_j \geq 1, \forall ij \in E$$

$$0 \leq x_i \leq 1, \forall i \in V$$

Avantage : relaxation lineaire soluble en temps polynomial donne une solution reelle avec un meilleur objectif... mais non admissible.

Algo d'approx : On va utiliser cette solution reelle non admissible pour en faire une solution entiere avec ratio d'approx garanti.

Arrondi de la solution réelle

- on résout la relaxation linéaire en temps polynomial :
on obtient une solution réelle minimum $\tilde{S} = (\tilde{x}_i)_{1 \leq i \leq n}$

Arrondi de la solution réelle

- on résout la relaxation linéaire en temps polynomial :
on obtient une solution réelle minimum $\tilde{S} = (\tilde{x}_i)_{1 \leq i \leq n}$
- on construit une solution entière $S = (x_i)_{1 \leq i \leq n}$ par arrondi :
 $x_i = 1$ ssi $\tilde{x}_i \geq 1/2$

Arrondi de la solution réelle

- on résout la relaxation linéaire en temps polynomial :
on obtient une solution réelle minimum $\tilde{S} = (\tilde{x}_i)_{1 \leq i \leq n}$
- on construit une solution entière $S = (x_i)_{1 \leq i \leq n}$ par arrondi :
 $x_i = 1$ ssi $\tilde{x}_i \geq 1/2$
- on montre que S est un vertex cover et que $w(S) \leq 2w(S^*)$,
ou S^* est une solution minimum en nombre entiers

Arrondi de la solution réelle

- on résout la relaxation linéaire en temps polynomial :
on obtient une solution réelle minimum $\tilde{S} = (\tilde{x}_i)_{1 \leq i \leq n}$
- on construit une solution entière $S = (x_i)_{1 \leq i \leq n}$ par arrondi :
 $x_i = 1$ ssi $\tilde{x}_i \geq 1/2$
- on montre que S est un vertex cover et que $w(S) \leq 2w(S^*)$,
ou S^* est une solution minimum en nombre entiers

Lemme

Soit S^ un MVC de poids minimum et soit S la solution obtenue par arrondi à partir de la solution minimum \tilde{S} de la relaxation linéaire. Alors S est un vertex cover et $w(S) \leq 2w(S^*)$.*

Preuve du ratio d'approximation

Lemme

Soit S^ un MVC de poids minimum et soit S la solution obtenue par arrondi a partir de la solution minimum \tilde{S} de la relaxation lineaire. Alors S est un vertex cover et $w(S) \leq 2w(S^*)$.*

Démonstration.

Preuve du ratio d'approximation

Lemme

Soit S^ un MVC de poids minimum et soit S la solution obtenue par arrondi a partir de la solution minimum \tilde{S} de la relaxation lineaire. Alors S est un vertex cover et $w(S) \leq 2w(S^*)$.*

Démonstration.

- on remarque que $w(\tilde{S}) \leq w(S^*)$
car la relaxation lineaire est moins contrainte

Preuve du ratio d'approximation

Lemme

Soit S^* un MVC de poids minimum et soit S la solution obtenue par arrondi a partir de la solution minimum \tilde{S} de la relaxation lineaire. Alors S est un vertex cover et $w(S) \leq 2w(S^*)$.

Démonstration.

- on remarque que $w(\tilde{S}) \leq w(S^*)$
car la relaxation lineaire est moins contrainte
- on montre que S est bien un vertex cover
 - ▶ soit $ij \in E$, comme $\tilde{x}_i + \tilde{x}_j \geq 1$, alors $\tilde{x}_i \geq 1/2$ ou $\tilde{x}_j \geq 1/2$
 - ▶ on a donc $x_i = 1$ ou $x_j = 1$: S est un vertex cover

Preuve du ratio d'approximation

Lemme

Soit S^* un MVC de poids minimum et soit S la solution obtenue par arrondi a partir de la solution minimum \tilde{S} de la relaxation lineaire. Alors S est un vertex cover et $w(S) \leq 2w(S^*)$.

Démonstration.

- on remarque que $w(\tilde{S}) \leq w(S^*)$
car la relaxation lineaire est moins contrainte
- on montre que S est bien un vertex cover
 - ▶ soit $ij \in E$, comme $\tilde{x}_i + \tilde{x}_j \geq 1$, alors $\tilde{x}_i \geq 1/2$ ou $\tilde{x}_j \geq 1/2$
 - ▶ on a donc $x_i = 1$ ou $x_j = 1$: S est un vertex cover
- on montre que $w(S) \leq 2w(\tilde{S})$
 - ▶ comme $x_i = 1$ si $\tilde{x}_i \geq 1/2$ et $x_i = 0$ sinon, alors $x_i \leq 2\tilde{x}_i$
 - ▶ d'ou, $w(S) \leq 2w(\tilde{S})$

Preuve du ratio d'approximation

Lemme

Soit S^* un MVC de poids minimum et soit S la solution obtenue par arrondi a partir de la solution minimum \tilde{S} de la relaxation lineaire. Alors S est un vertex cover et $w(S) \leq 2w(S^*)$.

Démonstration.

- on remarque que $w(\tilde{S}) \leq w(S^*)$
car la relaxation lineaire est moins contrainte
- on montre que S est bien un vertex cover
 - ▶ soit $ij \in E$, comme $\tilde{x}_i + \tilde{x}_j \geq 1$, alors $\tilde{x}_i \geq 1/2$ ou $\tilde{x}_j \geq 1/2$
 - ▶ on a donc $x_i = 1$ ou $x_j = 1$: S est un vertex cover
- on montre que $w(S) \leq 2w(\tilde{S})$
 - ▶ comme $x_i = 1$ si $\tilde{x}_i \geq 1/2$ et $x_i = 0$ sinon, alors $x_i \leq 2\tilde{x}_i$
 - ▶ d'ou, $w(S) \leq 2w(\tilde{S})$
- comme $w(\tilde{S}) \leq w(S^*)$, on obtient $w(S) \leq 2w(S^*)$