Programmation Linéaire en nombres entiers
MOD 4.4: Recherche opérationnelle

Nicolas Bousquet

Ecole Centrale de Lyon

Rappel

Lors des premiers cours, on a vu:

- Comment résoudre un programme linéaire en nombre réel grâce au Simplexe.
- Comment définir le dual d'un PL.
- Que dans le cas d'un programme linéaire en nombre entier (PLNE), c'était plus compliqué.
- On a vu (en BE), que si la matrice de contraintes est TU alors solutions réelles et entières étaient "les mêmes".

Aujourd'hui:
Que fait-on dans les autres cas?

Algorithm “Branch and Bound”

Idée:
Si une solution n’est pas entière alors on va essayer de rendre un coefficient “entier”.

Algorithm "Branch and Bound"

Idée:
Si une solution n'est pas entière alors on va essayer de rendre un coefficient "entier".

$\Rightarrow$ On prend un coefficient x_i non entier dans la solution optimale courante x tel que x_i est non entier. On crée deux PL, un où on rajoute la contrainte $x_i \leq \lfloor x_i \rfloor$ and l'autre $x_i \geq \lceil x_i \rceil$.
On fait deux appels récursifs et on renvoie la meilleure solution.

Algorithm "Branch and Bound"

Idée:
Si une solution n'est pas entière alors on va essayer de rendre un coefficient "entier".

$\Rightarrow$ On prend un coefficient x_i non entier dans la solution optimale courante x tel que x_i est non entier. On crée deux PL, un où on rajoute la contrainte $x_i \leq \lfloor x_i \rfloor$ and l'autre $x_i \geq \lceil x_i \rceil$.
On fait deux appels récursifs et on renvoie la meilleure solution.

Structure de l'algorithme:
On veut résoudre un PL en nombre entiers:

- On commence par le résoudre en nombre réel.

Algorithm "Branch and Bound"

Idée:
Si une solution n'est pas entière alors on va essayer de rendre un coefficient "entier".

$\Rightarrow$ On prend un coefficient x_i non entier dans la solution optimale courante x tel que x_i est non entier. On crée deux PL, un où on rajoute la contrainte $x_i \leq \lfloor x_i \rfloor$ and l'autre $x_i \geq \lceil x_i \rceil$.
On fait deux appels récursifs et on renvoie la meilleure solution.

Structure de l'algorithme:
On veut résoudre un PL en nombre entiers:

- On commence par le résoudre en nombre réel.
- Si la solution optimale est entière on renvoie la solution et sa valeur.

Algorithm "Branch and Bound"

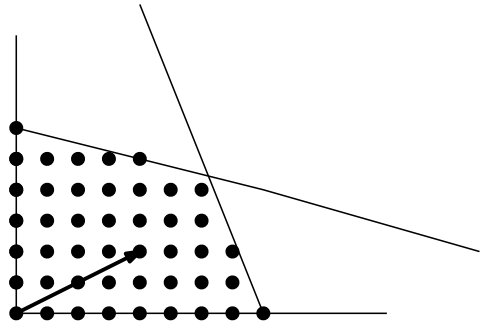
Idée:
Si une solution n'est pas entière alors on va essayer de rendre un coefficient "entier".

$\Rightarrow$ On prend un coefficient x_i non entier dans la solution optimale courante x tel que x_i est non entier. On crée deux PL, un où on rajoute la contrainte $x_i \leq \lfloor x_i \rfloor$ and l'autre $x_i \geq \lceil x_i \rceil$.
On fait deux appels récursifs et on renvoie la meilleure solution.

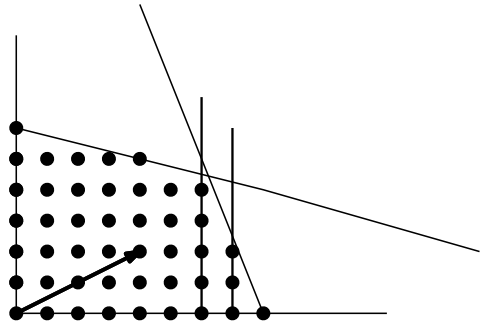
Structure de l'algorithme:
On veut résoudre un PL en nombre entiers:

- On commence par le résoudre en nombre réel.
- Si la solution optimale est entière on renvoie la solution et sa valeur.
- Sinon on prend une variable non entière $x = \alpha$. On crée deux nouveaux PL avec les anciennes contraintes plus $x \leq \lfloor \alpha \rfloor$ pour l'un et $x \geq \lceil \alpha \rceil$ pour l'autre.
  On retourne le maximum des deux résultats.

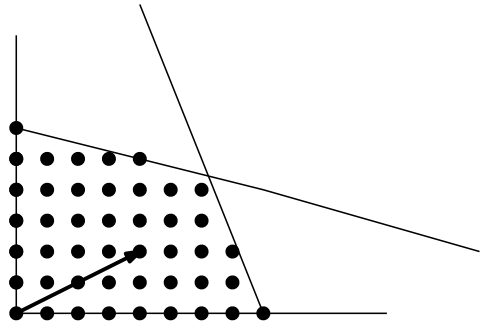
Illustration

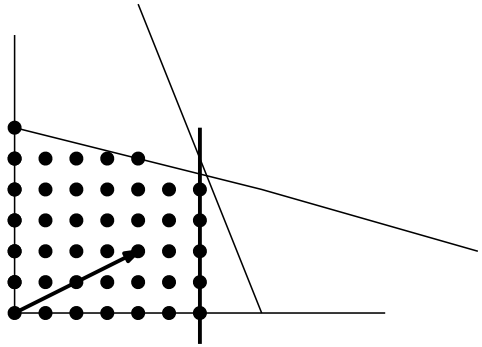


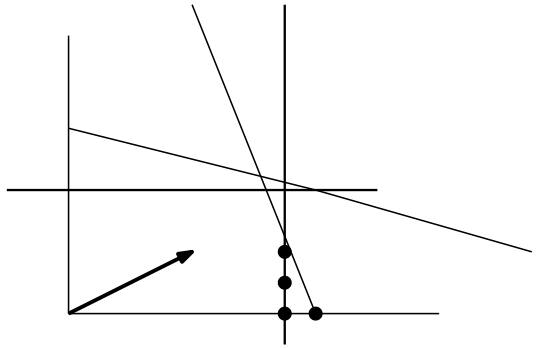
Illustration



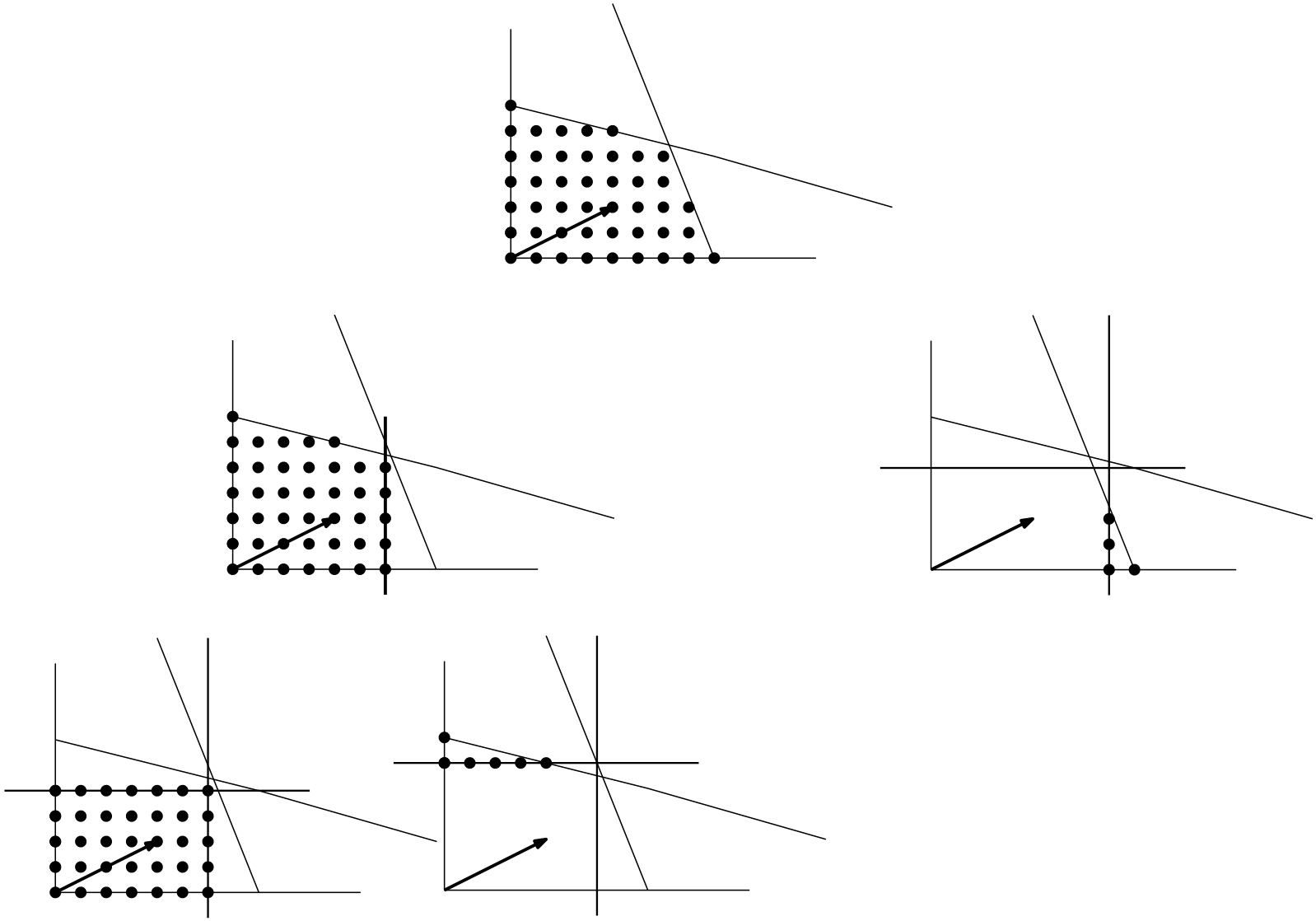
Illustration



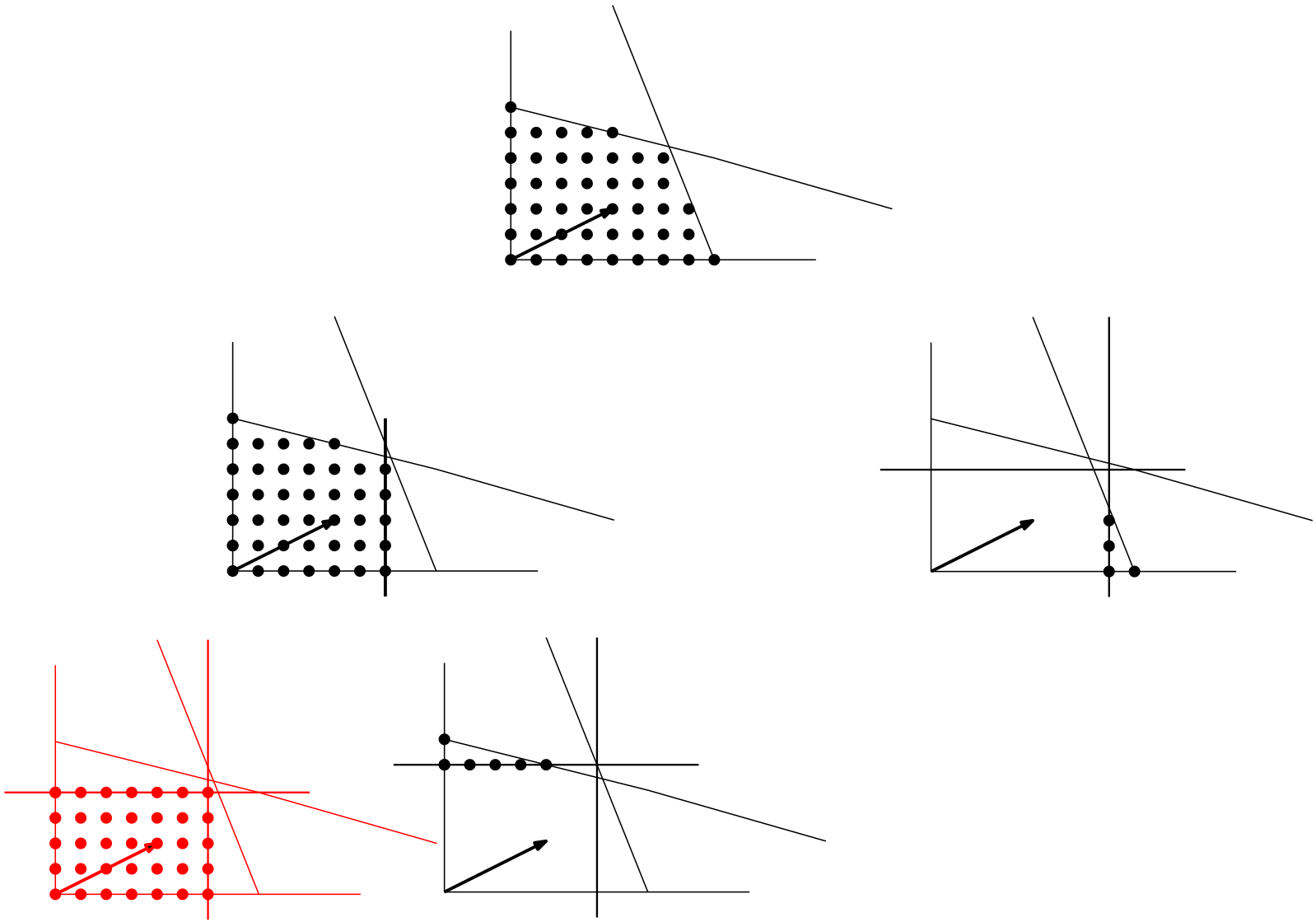




Illustration



Illustration



Formalisation

Soit (P) le programme linéaire en entrée.

- On calcule une solution optimale x^* de (P).
- Si le PL n'est pas satisfiable, alors on renvoie $-\infty$ (ou $+\infty$ pour un problème de minimisation).
- Si x^* est un vecteur entier, alors on renvoie x^*.
- Sinon soit x_i une coordonnée de x^* non entière
- Créer deux PL (P') et (P") où (P')= (P) plus la contrainte $x_i \leq \lfloor x_i \rfloor$ et (P") = (P) plus la contrainte $x_i \geq \lceil x_i \rceil$.
- Renvoyer le maximum entre l'optimal entier de (P') et l'optimal entier de (P").

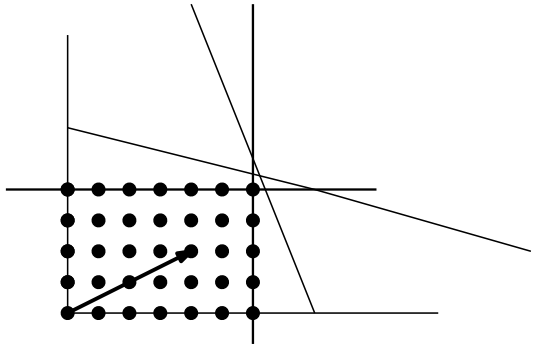
Lemme: L'algorithme termine.

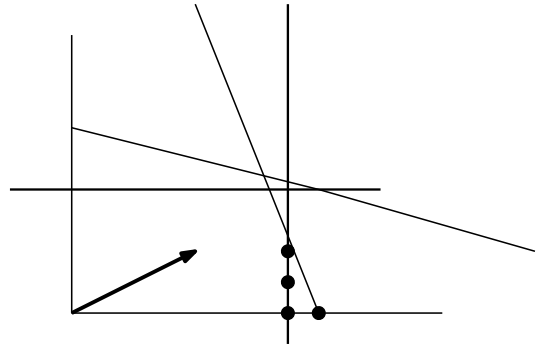
Améliorations possibles

Remarque:

On peut se souvenir de la meilleure solution entière qu'on a trouvé jusqu'à présent.

Si la solution en nombre réel de la composante qu'on considère est inférieure, on peut "couper la branche".



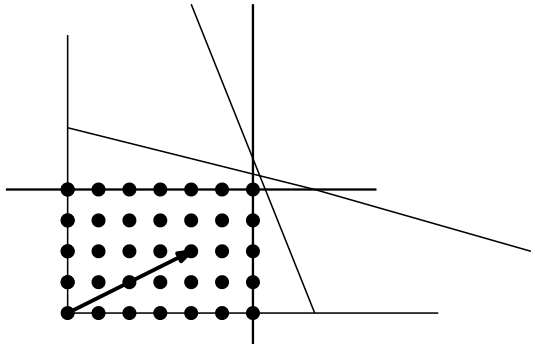


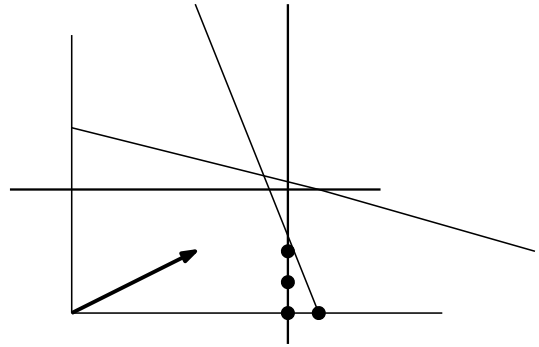
Améliorations possibles

Remarque:

On peut se souvenir de la meilleure solution entière qu'on a trouvé jusqu'à présent.

Si la solution en nombre réel de la composante qu'on considère est inférieure, on peut "couper la branche".





Remarque 2:

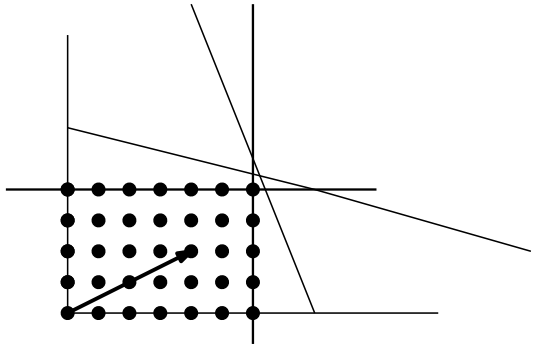
- On peut en fait facilement calculer la solution optimale du nouveau PL *sans tout recalculer* !
- Bien choisir son heuristique pour la variable à modifier (vaste sujet)

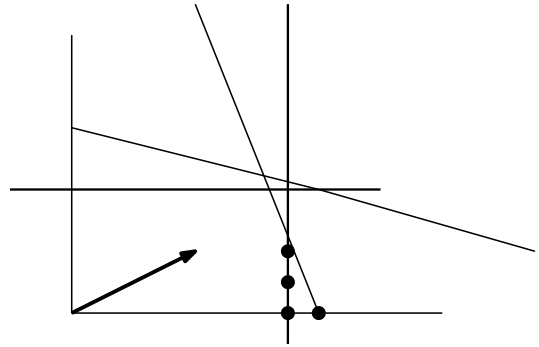
Améliorations possibles

Remarque:

On peut se souvenir de la meilleure solution entière qu'on a trouvé jusqu'à présent.

Si la solution en nombre réel de la composante qu'on considère est inférieure, on peut "couper la branche".





Remarque 2:

- On peut en fait facilement calculer la solution optimale du nouveau PL *sans tout recalculer* !
- Bien choisir son heuristique pour la variable à modifier (vaste sujet)

Remarque 3:

Si on considère un problème $\{0, 1\}$ (où toutes les variables prennent des valeurs dans $0 - 1$), alors les contraintes qu'on rajoute sont du type "je selectionne / je ne selectionne pas" la variable, ce qui est assez peu informatif...

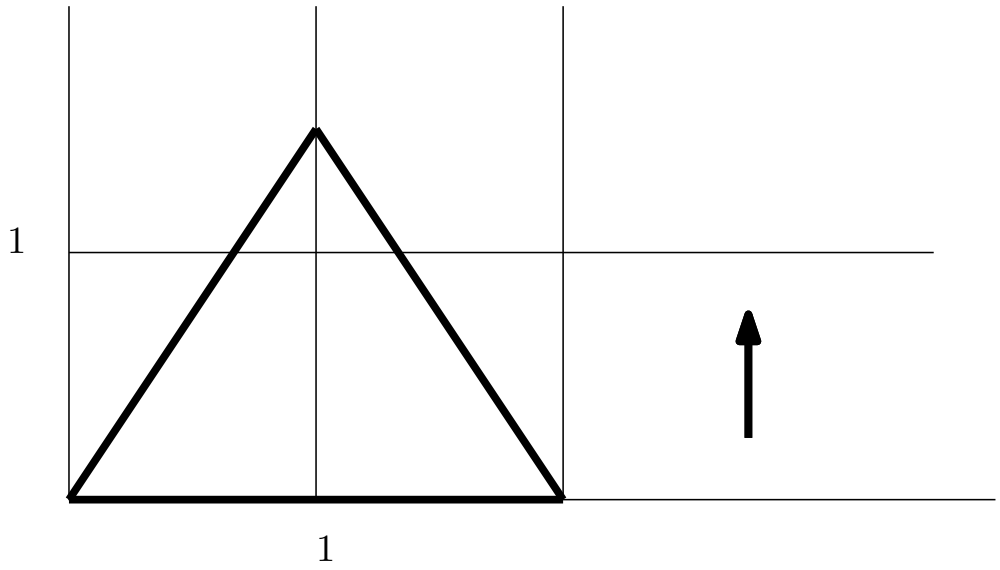
Méthode 2: Cutting planes

Idée: Trouver une nouvelle contrainte qui:

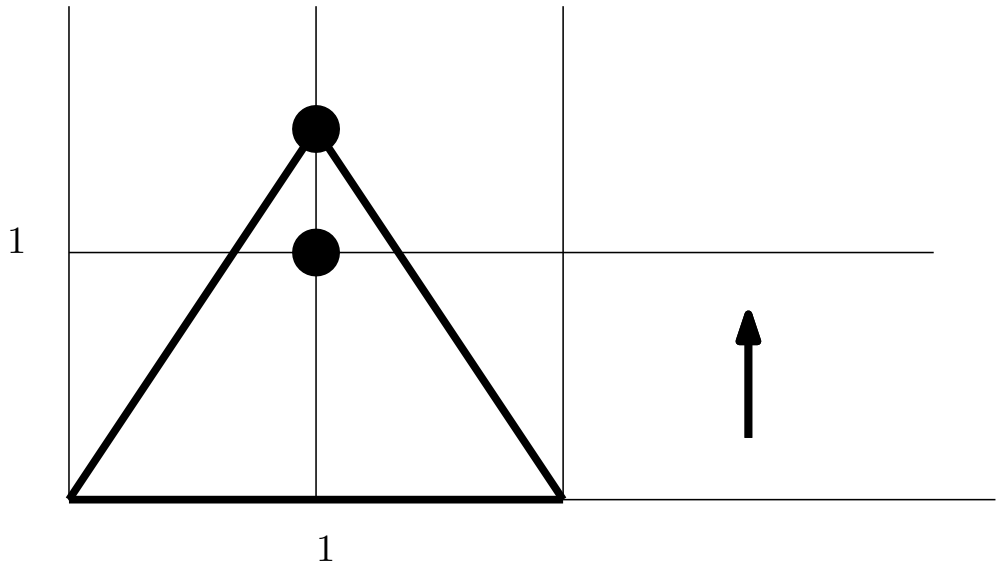
- Ne modifie pas l'ensemble des contraintes entières;
- Elimine un point non entier de l'emble des solutions du PL en nombre réel.

En d'autres termes, on cherche une contrainte qui ne va pas modifier les solutions entières (et donc l'optimale ne va pas changer) mais va couper une partie de l'ensemble faisable en nombre réel.

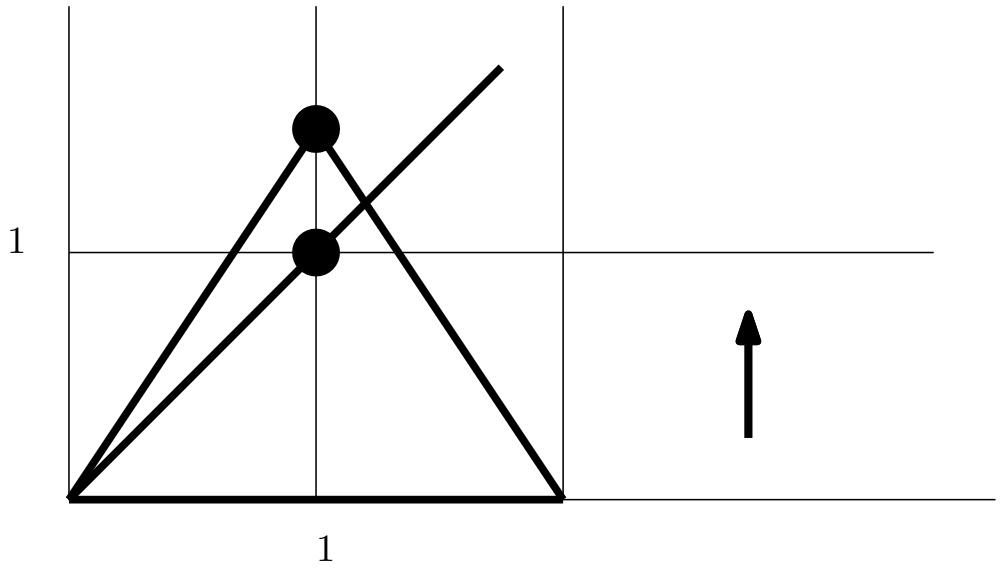
Exemple



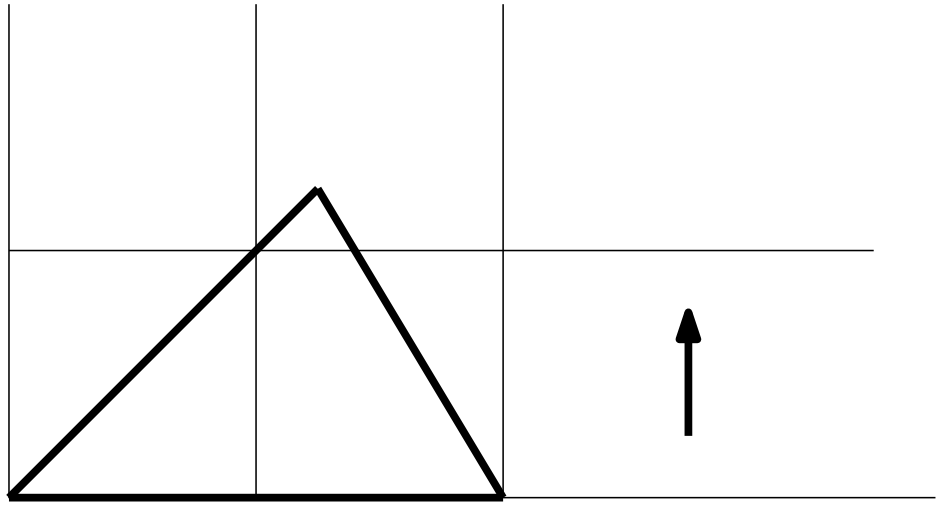
Exemple



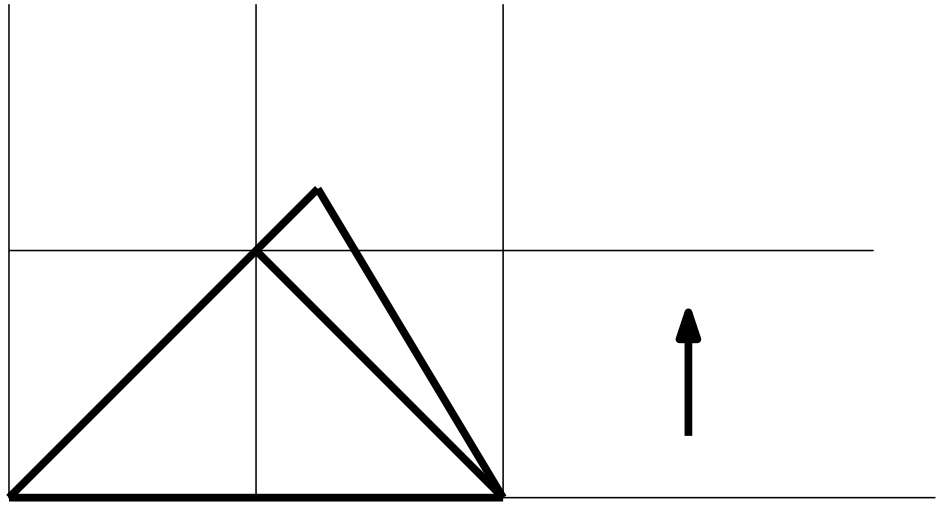
Exemple



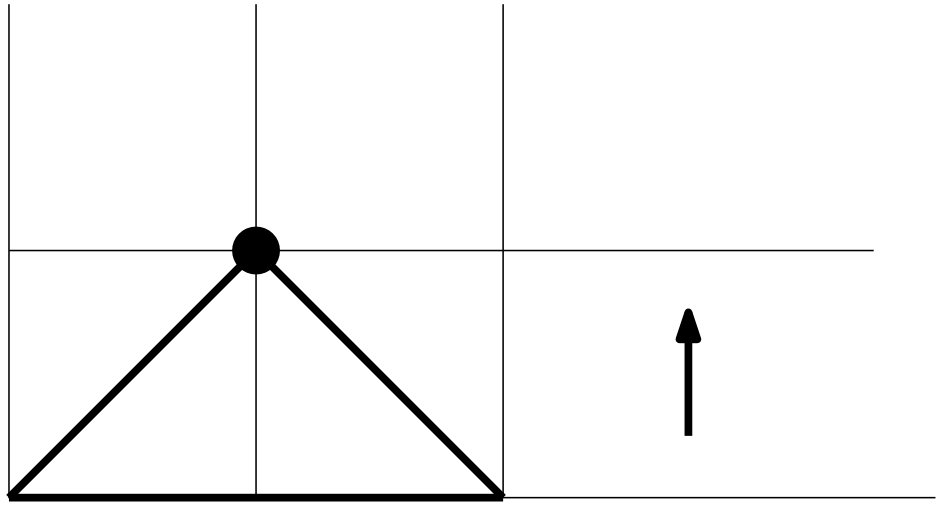
Exemple



Exemple



Exemple



Coupes de Gomory

Question: Comment trouver une telle coupe en général?

Coupes de Gomory

Question: Comment trouver une telle coupe en général?

- Soit le Simplexe renvoie un solution entière ✓.
- Soit il existe au moins un coefficient non entier (disons x_i) à la fin de l'algorithme. On a donc:

$$x_i + \sum_{j \text{ non basique}} a_j x_j = b_i$$

Coupes de Gomory

Question: Comment trouver une telle coupe en général?

- Soit le Simplexe renvoie un solution entière ✓.
- Soit il existe au moins un coefficient non entier (disons x_i) à la fin de l'algorithme. On a donc:

$$x_i + \sum_{j \text{ non basique}} a_j x_j = b_i$$

$$\Rightarrow x_i + \sum_{j \text{ non basique}} \lfloor a_j \rfloor x_j \leq b_i$$

(vrai pour n'importe quelle solution)

Coupes de Gomory

Question: Comment trouver une telle coupe en général?

- Soit le Simplexe renvoie un solution entière ✓.
- Soit il existe au moins un coefficient non entier (disons x_i) à la fin de l'algorithme. On a donc:

$$x_i + \sum_{j \text{ non basique}} a_j x_j = b_i$$

$$\Rightarrow x_i + \sum_{j \text{ non basique}} \lfloor a_j \rfloor x_j \leq b_i$$

(vrai pour n'importe quelle solution)

$$\Rightarrow x_i + \sum_{j \text{ non basique}} \lfloor a_j \rfloor x_j \leq \lfloor b_i \rfloor$$

(vrai pour n'importe quelle solution entière !)

Coupes de Gomory (cont.)

On rajoute la contrainte

$$x_i + \sum_{j \text{ non basique}} \lfloor a_j \rfloor x_j \leq \lfloor b_i \rfloor$$

au systême.

Coupes de Gomory (cont.)

On rajoute la contrainte

$$x_i + \sum_{j \text{ non basique}} \lfloor a_j \rfloor x_j \leq \lfloor b_i \rfloor$$

au systême.

Remarques:

- Toute solution entière satisfait le nouveau système.
- L'ancienen solution optimale ne satisfait pas la nouvelle contrainte.

Est-ce la meilleure chose à faire?
Pas forcément. Si le problème a une structure forte, il peut être plus intéressant de chercheur d'autres plans coupants...

Méthode 3: Branch and Cut

"Le meilleur des deux mondes"

- Il généralise à la fois "Branch and Bound" et "Cutting planes".
- C'est celui qui est en général implémenté dans les solveurs existants.

Partie 2

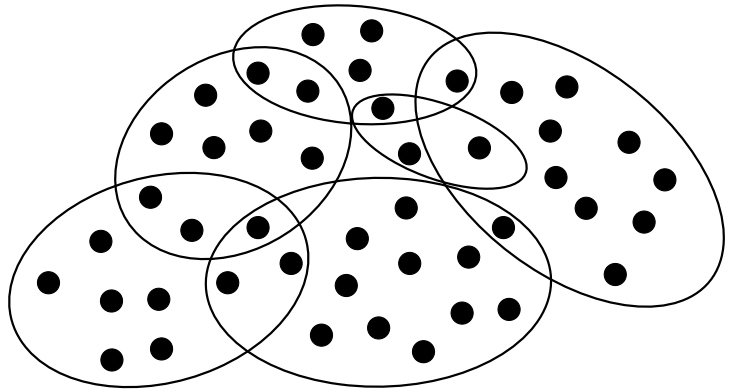
Écart entre valeurs optimales du primal et du dual pour les PLNE.

Hypergraphe

Définition (hypergraphe)

Un hypergraphe $H = (V, E)$ est une paire où:

- V est un ensemble de sommets.
- E est un ensemble de **sous ensembles** de sommets appelés hyperarêtes.



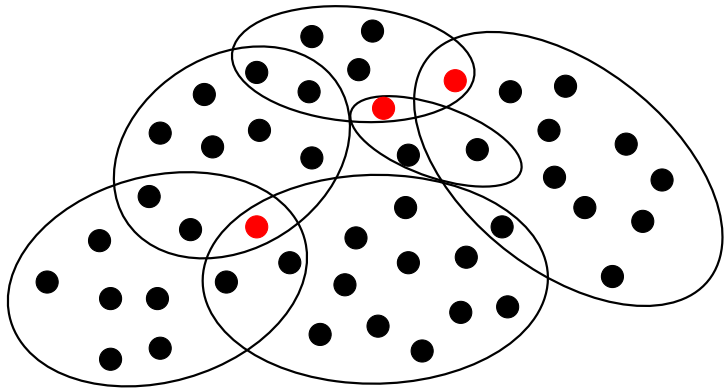
Utilisations:

- Quand on veut modéliser des relations qui ne sont pas binaires mais d'arité arbitrairement grande (ex. modéliser des groupes dans les réseaux sociaux).
- Généralisent et modélisent de nombreux problèmes de graphes.

Transversal d'hypergraphe

Définition (transversal)

Un transversal d'un hypergraphe $H = (V, E)$ est un ensemble de sommets qui contient au moins un sommet de chaque hyperarête.



Exemple:

- Si les hyperarêtes sont les arêtes d'un graphe alors un transversal est un ensemble de sommets qui permet de surveiller toutes les arêtes.
- Si les hyperarêtes sont les st-chemins dans le graphe, alors un transversal est exactement une coupe minimale.

Transversal minimum et programme linéaire

Soit $H = (V, E)$ un hypergraphe.

- Créons une variable x_v pour chaque sommet de l'hypergraphe.
- Le but est de minimiser la taille d'un transversal:

$$\tau = \min \sum_{v \in V} x_v.$$

- Pour chaque hyperarête on veut sélectionner au moins un sommet. Pour toute hyperarête $e \in E$:

$$\sum_{v \in e} x_v \geq 1.$$

- On veut soit prendre, soit ne pas prendre un sommet. Pour tout $v \in V$

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

Écart d'intégralité

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$$\begin{array}{ll} \sum_{v \in e} x_v \geq 1 & \text{pour tout } e \in E \\ x_v \in \{0,1\} & \text{pour tout } v \in V \end{array}$$

Écart d'intégralité

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$$\sum_{v \in e} x_v \geq 1 \quad \text{pour tout } e \in E$$
$$x_v \in \{0, 1\} \quad \text{pour tout } v \in V$$

$$\tau^* = \min \sum_{v \in V} x_v$$

sous les contraintes:

$$\sum_{v \in e} x_v \geq 1 \quad \text{pour tout } e \in E$$
$$0 \leq x_v \leq 1 \quad \text{pour tout } v \in V$$

Le PL de droite s'appelle la relaxation fractionnaire du PL de gauche.
La valeur optimale de la relaxation fractionnaire est notée avec une $*$.

Écart d'intégralité

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$x_v \in \{0, 1\}$ pour tout $v \in V$

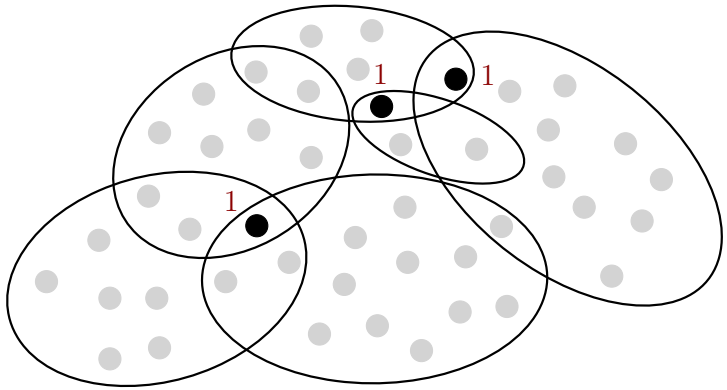
$$\tau^* = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$0 \leq x_v \leq 1$ pour tout $v \in V$

Le PL de droite s'appelle la relaxation fractionnaire du PL de gauche.
La valeur optimale de la relaxation fractionnaire est notée avec une $*$.



Écart d'intégralité

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$x_v \in \{0, 1\}$ pour tout $v \in V$

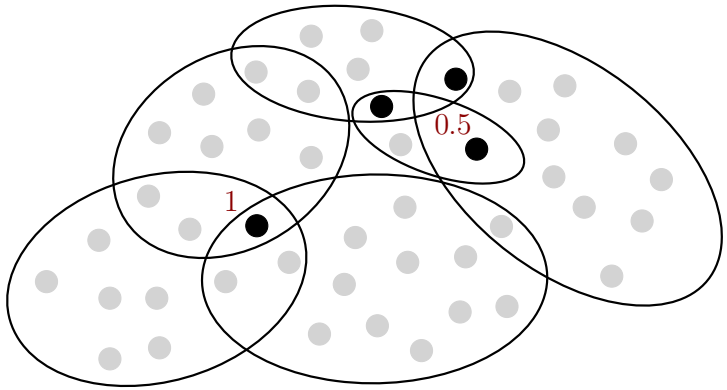
$$\tau^* = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$0 \leq x_v \leq 1$ pour tout $v \in V$

Le PL de droite s'appelle la relaxation fractionnaire du PL de gauche.
La valeur optimale de la relaxation fractionnaire est notée avec une $*$.



Écart d'intégralité

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$x_v \in \{0, 1\}$ pour tout $v \in V$

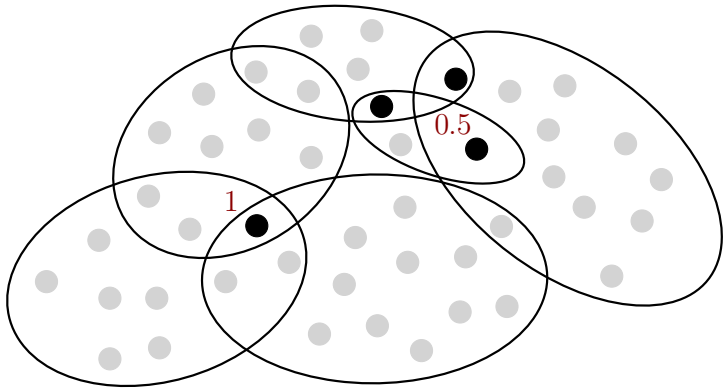
$$\tau^* = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$0 \leq x_v \leq 1$ pour tout $v \in V$

Le PL de droite s'appelle la relaxation fractionnaire du PL de gauche.
La valeur optimale de la relaxation fractionnaire est notée avec une $*$.



L'écart entre l'optimal du programme linéaire d'origine et sa relaxation fractionnaire est appelé l'écart d'intégralité.

Question: Peut-on borner l'écart d'intégralité?

NON !

NON !

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$$\sum_{v \in e} x_v \geq 1 \quad \text{pour tout } e \in E$$
$$x_v \in \{0, 1\} \quad \text{pour tout } v \in V$$

$$\tau^* = \min \sum_{v \in V} x_v$$

sous les contraintes:

$$\sum_{v \in e} x_v \geq 1 \quad \text{pour tout } e \in E$$
$$0 \leq x_v \leq 1 \quad \text{pour tout } v \in V$$

Exemple:

- $V = \{1, \ldots, 2n-1\}$.
- $e \in E$ si et seulement si $|e| = n$.

$\tau \geq n$.

Par contradiction. Si on ne sélectionne que $n-1$ sommets, il reste n sommets dans le complémentaire. Comme tous les ensembles de taille n sont des hyperarêtes, une hyperarête n'est pas touchée, contradiction.

$\tau^* < 2$.

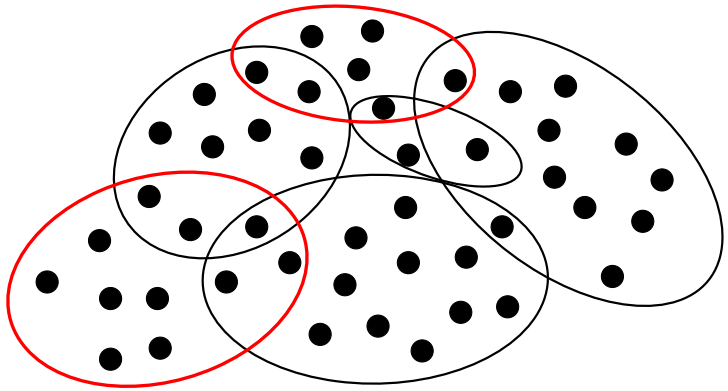
Donner un poids de $\frac{1}{n}$ à tous les sommets (i.e. poser $x_v = \frac{1}{n}$ pour tout sommet). Comme toutes les hyperarêtes ont taille n, le poids de chaque hyperarête vaut 1.

$\Rightarrow$ L'écart d'intégralité vaut au moins $n/4$.

Packing

Définition (packing)

Un packing dans un hypergraphe est un ensemble d'hyperarêtes deux à deux disjointes.



Exemple:

- Si les hyperarêtes sont les *st*-chemins, il s'agit de chemins disjoints.
- Si les hyperarêtes sont des activités, il s'agit d'activités que l'on peut programmer en parallèle (aucune personne ne souhaite participer à plusieurs activités).

Expression comme un problème linéaire

- On crée une variable y_e pour chaque hyperarête.
- On désire maximiser le nombre d'hyperarêtes sélectionnées:

$$\nu = \max \sum_{e \in E} y_e$$

- Chaque sommet apparaît dans au plus une hyperarête du packing. Pour tout sommet:

$$\sum_{e/v \in e} y_e \leq 1.$$

- Chaque hyperarête est soit sélectionnée, soit non-sélectionnée. Pour toute hyperarête e,

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

Donc le problème du packing maximal peut s'exprimer comme un programme linéaire en nombres entiers.

$$\nu = \max \sum_{e \in E} y_e$$

$$\text{sous les contraintes} \quad \begin{array}{ll} \sum_{e/v \in e} y_e \leq 1 & \text{pour tout } v \in V \\ y_e \in \{0, 1\} & \text{pour tout } e \in E \end{array}$$

Tranversal et packing

$$\tau = \min \sum_{v \in V} x_v$$

sous les contraintes:

$\sum_{v \in e} x_v \geq 1$ pour tout $e \in E$

$x_v \in \{0, 1\}$ pour tout $v \in V$

$$\nu = \max \sum_{e \in E} y_e$$

sous les contraintes:

$\sum_{e/v \in e} y_e \leq 1$ pour tout $v \in V$

$y_e \in \{0, 1\}$ pour tout $e \in E$

Quel est le dual du problème du packing maximal?

- On crée une nouvelle variable z_v pour chaque contrainte, i.e. pour chaque sommet v.
- Le problème dual est un problème de minimisation.
- Le vecteur objectif du dual est le vecteur composé de 1 car c'est le vecteur contrainte du primal.
- Le vecteur de contraintes du dual est le vecteur composé de 1 car c'est le vecteur objectif du primal.
- On suppose que dans le dual les variables vaudront aussi 0 ou 1.

Exercice: Vérifiez que problème est celui du transversal minimum !

Dual d'un PL en nombre entier

Nous savons ce qu'est un dual pour un PL en nombre réel, et nous venons de le définir pour un PL en nombre entier "avec les mains".

Définition:
Deux programmes linéaires en nombre entiers sont duaux l'un de l'autre si et seulement si leurs relaxations fractionnaires respectives sont duales l'une de l'autre.

Exercice:
Montrer formellement que le problème du transversal minimum et celui du packing maximum sont duaux l'un de l'autre.

Écart primal-dual en nombre entier

L'écart entre la valeur optimale du primal et la valeur optimale du dual peut être arbitrairement grande.

Exemple:

- $V = \{1, \ldots, 2n-1\}$.
- $e \in E$ si et seulement si $|e| = n$.

Aucun transversal n'a une taille plus petite que n.
On l'a déjà vu.

Il n'existe pas de packing de taille 2.
Deux hyperarêtes contiennent $2n$ sommets. Comme le nombre total de sommets est $2n-1$, toute les hyperarêtes s'intersectent.

En fait, on a le résultat plus général suivant. Pour tout hypergraphe H, on a:

$$\nu(H) \leq \nu^*(H) = \tau^*(H) \leq \tau(H)$$

$$\text{Primal (max)} \leq \text{ Primal}^* = \text{ Dual}^* \leq \text{ Dual (min)}$$

Mathématiques discrètes vs continues

Les PL en nombre entiers sont des problèmes qui sont, tant d'un point de vue algorithmique que combinatoire extrememement difficile.
Il faut donc en général avoir une structure forte pour pouvoir exhiber de bonnes bornes en utilisant la PL.

Cela illustre l'écart de difficulté entre les mathématiques réelles (bien connues) par rapport aux mathématiques discrètes. En particulier on perd:

- Quasiment toute l'analyse.
- La notion d'ordre sur les éléments.