Programmation Linéaire en Nombres Entiers (PLNE)

- Toutes les variables sont obligées de prendre des valeurs entières

- En général les coefficients $a_{i,j}$ sont aussi entiers

- Et, donc, on peut se limiter à des constantes b_j entières

- Si, les variables sont aussi contraintes d'être entre 0 et 1, elles n'ont que les deux valeurs possibles 0 et 1 et on parle de $PL\ en\ 0-1$

Maintenant on peut exprimer des contraintes telles $x_1 \geq 1$ ou $x_2 \geq 1$
par la contrainte $x_1 + x_2 \geq 1$ parce que des "solutions" comme $x_1 = x_2 = 0.5$ sont interdites.

Le PL ordinaire avec les mêmes variables, contraintes et fonction objective s'appelle le *PL relaxé* du PLNE.
Sa solution optimale donne une borne (supérieure pour un problème de maximisation) sur la solution optimale du PLNE.

(Mais la solution optimale du PLNE peut être très différente ou même ne pas exister.)

Aspects Théoriques

- Chercher une solution réalisable du PLNE équivaut à chercher un point dans le polytope de solutions réalisables avec tous ses coordonnés entiers

- Si le polytope est long et étroit il n'est pas évident qu'il contient de tels points

- Décider si un PLNE possède des solutions réalisables est un exemple d'un problème NP-complet une classe de problèmes pour lesquels personne ne sait s'il existe ou non un algorithme efficace (opérant en temps polynomial)

- Les meilleurs algorithmes connus sont capables de traiter des programmes avec quelques dizaines de variables
(par comparaison, on peut traiter des PL de plusieurs milliers de variables)

- Méthodes heuristiques

Des cas où les deux programmes sont équivalents

- Pour quelques types de PL, on sait que (s'il existe une solution optimale,) il existe une solution optimale entière

- Donc, le PLNE et le PL ont la même valeur optimale
  et c'est beaucoup plus vite de résoudre le PL

- Essentiellement des problèmes sur des réseaux

- Des programmes où chaque variable est présente dans deux contraintes au maximum

- Et avec des coefficients de ± 1

Les Méthodes de "Séparation et Evaluation"

- On choisit une variable qui a une fourchette de valeurs possibles, disons $[min, max]$

- On construit deux nouveaux problèmes, un avec $[min, mid]$, et l'autre avec $[mid+1, max]$
  séparation

- On utilise la borne sur chaque problème fournie par son programme relaxé
  évaluation

- Si cette borne est inférieure (pour un problème de maximisation) à une solution déjà trouvée, terminer cette branche

- Et bien sûr teminer la branche si le programme relaxé n'est pas faisable

- Sinon continuer

A chaque moment il y a un arbre de problèmes à résoudre et la solution optimale est ou la meilleure solution déjà trouvée ou la meilleure de celles des problèmes contenus dans l'arbre

La Méthode de Dakin pour le PLNE général

- Parcourir l'arbre en profondeur (ce qui a une bonne chance de produire une solution réalisable assez vite)
- Choisir à chaque itération le problème possédant la meilleure solution de son programme relaxé
- Séparer sur une variable dont la valeur (dans la solution optimale du programme relaxé) est la plus proche d'un entier
- mid est cette valeur (arrondie)

Des problèmes sur les réseaux exprimés en programmes linéaires Classement

- **Affectation** Réseau trivial? ensemble de sources (disponibilité $= 1$) et de puits (demande $= 1$) et où chaque arc est d'une source à un puits

- **Transport** Demandes, disponibilités (et capacités des arcs) peuvent être supérieures à 1 mais toujours chaque arc est d'une source à un puits

- **Transbordement** Réseau général avec capacités infinies

- **Flot maximum** Réseau général avec capacités finies

- **Flot de coût minimum** Réseau général avec capacités finies et coûts différents sur les arcs

Ici on ne considère qu'un algorithme pour le problème de flot maximum.

L'algorithme de Ford et Fulkerson
Flot maximum

Il suffit de considérer le cas d'une source et un puits

Basé sur l'idée d'une chaîne augmentante: une suite de changements possibles $\pm\alpha$ dans une chaîne d'arcs dont le résultat est de transférer quantité α de la source à un autre sommet.

NB: l'augmentation dans une chaîne augmentante peut diminuer le flux dans certains arcs!

Facile de trouver des chaînes augmentantes dans un arbre enraciné à la source

Si on trouve une chaîne augmentante au puits, on peut augmenter le flot actuel (initialisé à 0) par le minimum des changements possibles sur les arcs de la chaîne

Sinon on a trouvé un flot maximum.

La coupe minimum

Quand l'algorithme ne trouve plus de chaîne augmentante au puits, il a trouvé un ensemble E de sommets autour de la source (ceux pour lesquels il existe une chaîne augmentante) tel que tous les arcs partant de E sont saturés (et tous ceux entrant en E ont flux nul);
dans le flot maximum trouvé le flux total sur ces arcs partant de E est le même que le flot total trouvé;
donc l'algorithme a trouvé une coupe avec cette valeur.
Théorème: cette coupe est minimum parmi les coupes séparant la source et le puits.
Du point de vue de la PL, la recherche d'une coupe minimum est le problème dual du celui du flot maximum!