

Programmation linéaire en nombres entiers

Alain Haït

Institut National Polytechnique de Toulouse



Septembre 2026

Plan du cours

1 Introduction

2 Résolution

3 Coupes

4 Recherche Arborescente

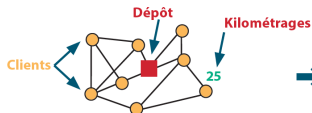
5 Solveurs

6 Modélisation

Optimisation combinatoire

Trouver un meilleur choix dans un ensemble fini (souvent très grand) de possibilités.

- Espace des solutions discontinu
- Rarement possible d'énumérer les solutions
- Problèmes généralement pas polynomiaux

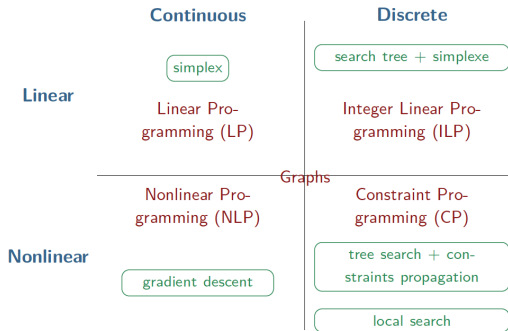


➔ **Objectif :**
Partant du dépôt,
visiter une seule fois
chacun des clients
et revenir au dépôt
en parcourant une
distance totale
minimale.

n	Nombre de possibilités	Temps de calcul par énumération
5	12	Micro-seconde
10	181440	Dixième de seconde
15	43 milliards	Dizaine d'heures
20	$60 \cdot 10^{15}$	Milliers d'années
25	$31 \cdot 10^{22}$	Milliards d'années

Optimisation combinatoire

- Algorithmes dédiés
- Méthodes de résolution génériques
- Hybrides



PL / PLNE

Programmation linéaire (PL)

- Variables de décision continues
- Problème polynomial
- Solveurs extrêmement performants

Programmation linéaire en nombres entiers (PLNE)

- Des variables entières ou binaires
- Problème NP-difficile dans le cas général
- Solveurs performants mais...

Objectifs

2 objectifs liés

- Savoir modéliser un problème d'optimisation en PL ou PLNE.
- Connaître les principes de résolution utilisés par les solveurs

Plus précisément

- Trouver les bons modèles pour résoudre efficacement les problèmes combinatoires auxquels on est confronté
- Élaborer la stratégie de résolution la plus performante possible en PLNE

Objectifs d'apprentissage

Programmation linéaire en nombres entiers

A l'issue des 10h de cours, vous devriez être capable de :

- Ess.** : Décrire de manière générique un programme linéaire en nombres entiers
- Ess.** : Modéliser un problème d'optimisation sous la forme d'un programme linéaire en nombres entiers
- Imp.** : Faire la différence entre la PLNE et la PL, notamment en termes de domaine des variables et de complexité
- Imp.** : Implémenter un modèle PLNE en Python
- Utl.** : Résoudre un programme linéaire en nombres entiers par séparation et évaluation
- Utl.** : Identifier des cas particuliers pour lesquels résoudre un programme linéaire en nombres entiers revient à résoudre sa relaxation linéaire

Plan du cours

1 Introduction

2 **Résolution**

3 Coupes

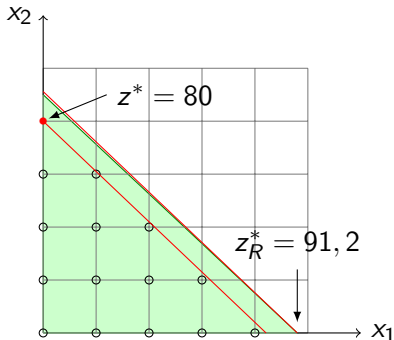
4 Recherche Arborescente

5 Solveurs

6 Modélisation

Résolution en PLNE

La solution de la relaxation linéaire ne donne pas l'optimum...



... mais on va l'utiliser pour guider la recherche.

Résolution en PLNE

La résolution a pour but

- de trouver l'optimum, et le **prouver**
- en un temps acceptable.

Elle s'appuie généralement sur une relaxation du problème

- qu'on sait résoudre facilement
- dont le résultat est le plus proche possible de celui du problème initial.

Relaxations

Soit un problème d'optimisation (P) $\max z = f(x)$ sous la contrainte $x \in X \subset \mathbb{R}^n$.

Définition

Le problème (R) $\max z_R = \varphi(x)$ sous la contrainte $x \in T \subset \mathbb{R}^n$ est une relaxation de (P) ssi $X \subseteq T$ et $\varphi(x) \geq f(x) \quad \forall x \in X$.

L'optimum z_R^* de la relaxation donne une **borne supérieure** de l'optimum de (P) .

Exemple

Problème (P) à résoudre :

$$\max z = x_1 + 2x_2$$

Sous

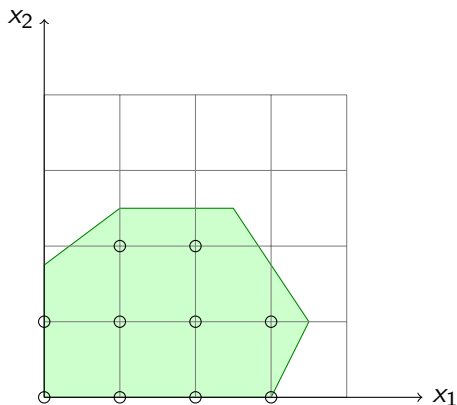
$$-3x_1 + 4x_2 \leq 7$$

$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1, x_2 \in \mathbb{N}$$



Exemple

(R) Relaxation linéaire
de (P) :

$$\max z_R = x_1 + 2x_2$$

Sous

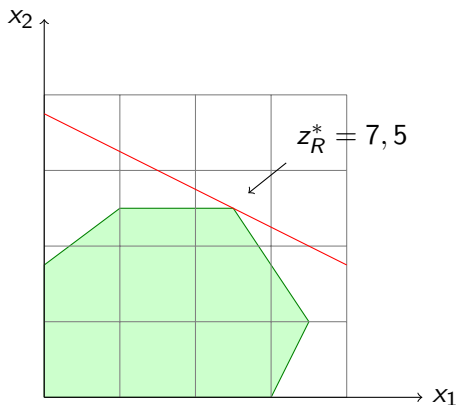
$$-3x_1 + 4x_2 \leq 7$$

$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1, x_2 \in \mathbb{R}^+$$



Relaxations

Propriété :

- Si (R) est une relaxation de (P) , alors $z_R^* \geq z^*$ (maximisation).

Qualité d'une relaxation

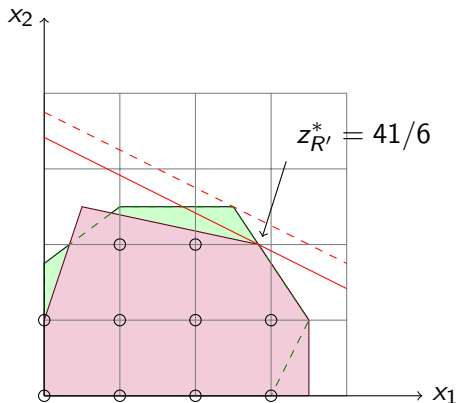
- Soient (R) et (R') deux relaxations de (P) . La relaxation (R') est **meilleure** que (R) si $z_{R'}^* \leq z_R^*$

C'est par exemple le cas si (R) est elle-même une relaxation de (R') .

- La relaxation (R) est **idéale** si $z_R^* = z^*$

Exemple

(P') Autre formulation
de (P) :



(R') relaxation linéaire de (P') est meilleure que (R).

Exemple

(P'') Autre formulation
de (P) :

$$\max z_{P''} = x_1 + 2x_2$$

Sous

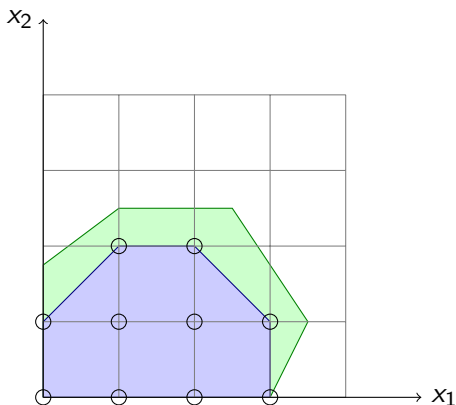
$$-x_1 + x_2 \leq 1$$

$$x_2 \leq 2$$

$$x_1 + x_2 \leq 4$$

$$x_1 \leq 3$$

$$x_1, x_2 \in \mathbb{N}$$



Exemple

(R'') relaxation linéaire
de (P'') :

$$\max z_{R'} = x_1 + 2x_2$$

Sous

$$-x_1 + x_2 \leq 1$$

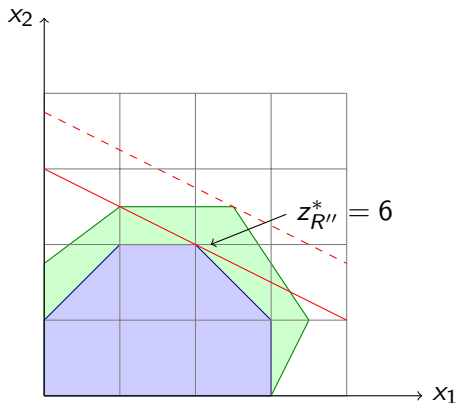
$$x_2 \leq 2$$

$$x_1 + x_2 \leq 4$$

$$x_1 \leq 3$$

$$x_1, x_2 \in \mathbb{R}^+$$

(P'') est une **formulation idéale** de (P).



Résolution d'un problème de PLNE

Recherche d'une relaxation linéaire telle que $z_R^* = z^*$.

- La résolution de la relaxation linéaire de (P) sur l'enveloppe convexe des solutions entières de P donne la solution optimale.
- Mais trouver cette enveloppe convexe est en soi un problème combinatoire difficile.

Résolution d'un problème de PLNE

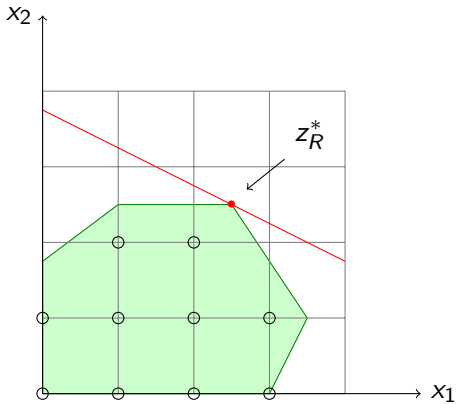
Stratégies de résolution :

- Coupes : ajout de contraintes au problème pour améliorer la relaxation
- Recherche arborescente : séparation successive de X en plusieurs sous-parties
- Stratégies hybrides
- Techniques de décomposition¹

1. Non traitées dans ce cours

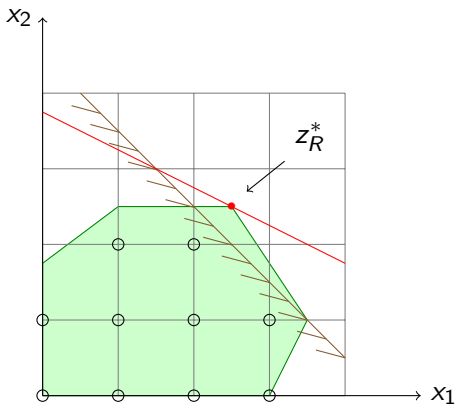
Coupes

Principe : ajout successif de contraintes au problème (P) pour renforcer sa relaxation (R).



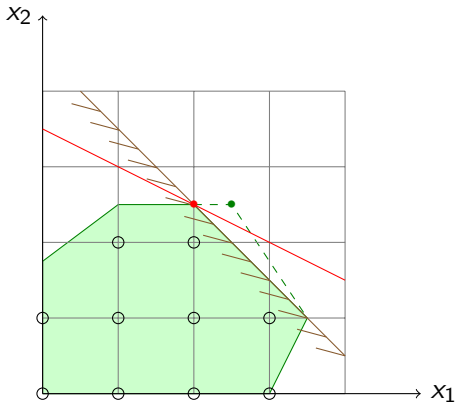
Coupes

Principe : ajout successif de contraintes au problème (P) pour renforcer sa relaxation (R).



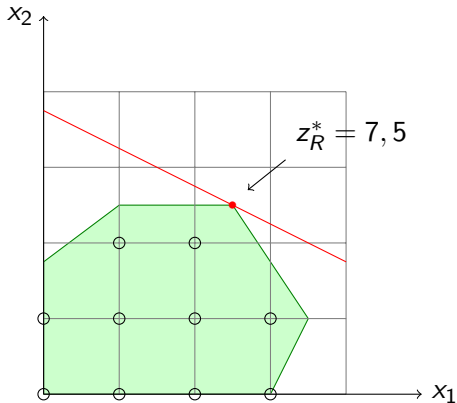
Coupes

Principe : ajout successif de contraintes au problème (P) pour renforcer sa relaxation (R).



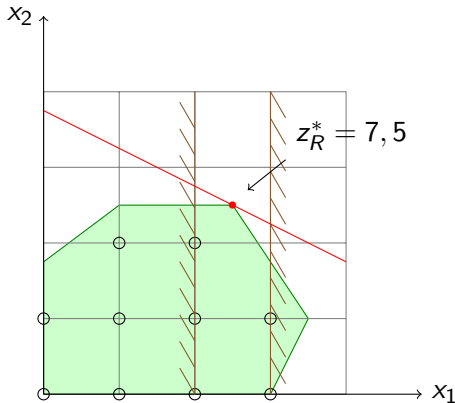
Recherche arborescente

Principe : séparation du problème et évaluation de la solution sur chaque sous-problème



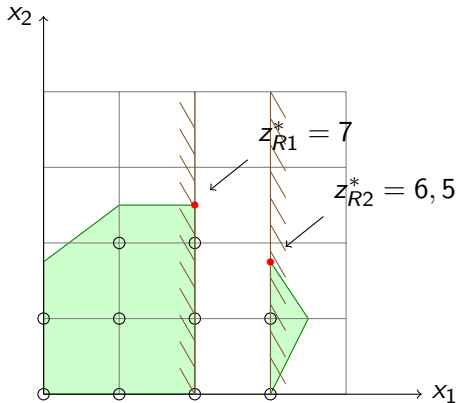
Recherche arborescente

Principe : séparation du problème et évaluation de la solution sur chaque sous-problème

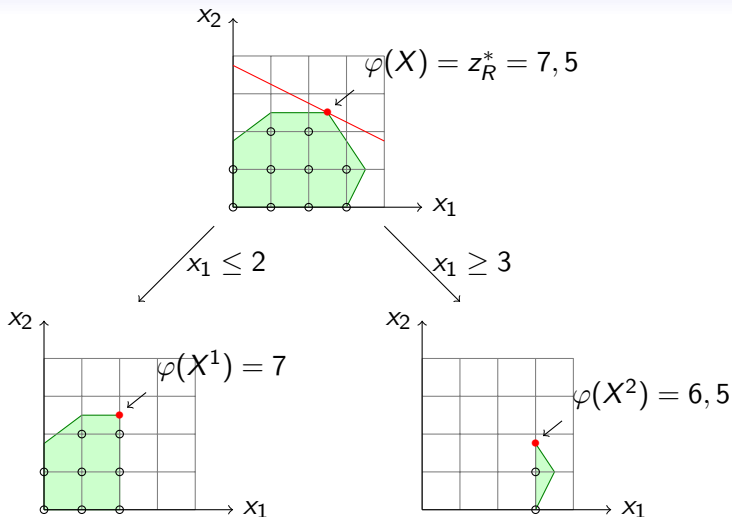


Recherche arborescente

Principe : séparation du problème et évaluation de la solution sur chaque sous-problème



Recherche Arborescente



Cas particulier

Question

Existe-t-il des cas pour lesquels on a directement l'enveloppe convexe des solutions entières (formulation idéale) ?

Cas particulier

Rappels de programmation linéaire

- Solutions admissibles : solutions du système $Ax = b$ avec $A(m \times n)$ matrice rectangulaire ($m < n$).
- Si $\text{Rang}(A) = m$, alors on peut trouver dans A une ou plusieurs sous-matrices carrées ($m \times m$) inversibles.
- Soit B une telle matrice. Les m vecteurs colonnes de B constituent une base de $\mathbb{R}^m$. Par abus de langage, on appelle B une base de A .
- la sous-matrice constituée par les colonnes restantes est appelée N .

Cas particulier

Rappels de programmation linéaire

- On peut écrire le système $Ax = b$ sous la forme $Bx_B + Nx_N = b$
- x_B sont les variables de base, x_N les variables hors base
- Le système $Ax = b$ est équivalent à $x_B = B^{-1}b + B^{-1}Nx_N$
- La solution $x_B = B^{-1}b$, $x_N = 0$ est appelée *solution de base* associée à B .

Cas particulier

Rappels de programmation linéaire

- On peut écrire le système $Ax = b$ sous la forme $Bx_B + Nx_N = b$
- x_B sont les variables de base, x_N les variables hors base
- Le système $Ax = b$ est équivalent à $x_B = B^{-1}b + B^{-1}Nx_N$
- La solution $x_B = B^{-1}b$, $x_N = 0$ est appelée *solution de base* associée à B .

Tout sommet du polyèdre des solutions admissibles correspond à une solution de base.

Cas particulier

Question

Existe-t-il des cas pour lesquels on a directement l'enveloppe convexe des solutions entières (formulation idéale) ?

On cherche des cas pour lesquels toutes les solutions de base correspondant à un sommet du polyèdre sont entières :

$$x_B = B^{-1}b, x_N = 0$$

Cas particulier

Question

Existe-t-il des cas pour lesquels on a directement l'enveloppe convexe des solutions entières (formulation idéale) ?

On cherche des cas pour lesquels toutes les solutions de base correspondant à un sommet du polyèdre sont entières :

$$x_B = B^{-1}b, x_N = 0$$

Propriété

Soit $B \in \mathbb{Z}^{m \times m}$ une matrice non singulière, alors $B^{-1}b$ est entier pour tout $b \in \mathbb{Z}^m$ si et seulement si $\text{Det}(B) = +\text{ ou } -1$.

Cas particulier

Définition

Une matrice $A \in \mathbb{Z}^{m \times n}$ de plein rang est **unimodulaire** si $\text{Det}(B) = +\text{ ou } - 1$ pour toute base B de A .

Corollaire

Si A est unimodulaire, il existe une solution optimale de la relaxation continue qui résout le problème en nombres entiers.

Mais : le nombre de bases peut être très grand (combinaisons de m parmi n).

Cas particulier

Définition

Une matrice est **totalelement unimodulaire** si chacune de ses sous-matrices a un déterminant égal à 0, 1 ou -1.

Conditions suffisantes pour être totalement unimodulaire :

- $a_{ij} \in \{-1; 0; 1\} \quad \forall i, j$
- $\sum_{i=1}^m |a_{ij}| \leq 2 \quad \forall j$
- Il existe une partition M_1, M_2 des indices des lignes telle que, si $\sum_{i=1}^m |a_{ij}| = 2$, alors $\sum_{i \in M_1} a_{ij} = \sum_{i \in M_2} a_{ij}$.

Cas particulier

Problème d'affectation : écrire la matrice pour une instance à 3 candidats et 3 postes. Vérifier qu'elle est totalement unimodulaire.

Plan du cours

1 Introduction

2 Résolution

3 **Coupes**

4 Recherche Arborescente

5 Solveurs

6 Modélisation

Coupes

Ajout successif de contraintes au problème pour se rapprocher de l'optimum.

Principe :

- L'ensemble des solutions entières de (P) est conservé
- La solution optimale courante (R) est exclue

Détermination de coupes :

- 1 Coupes d'arrondi à partir de l'objectif et des contraintes
- 2 Coupes spécifiques à un problème
- 3 Coupes généralisées : coupes de Gomory

Coupes d'arrondi

Problème (P) :

$$\max z = x_1 + 2x_2$$

Sous

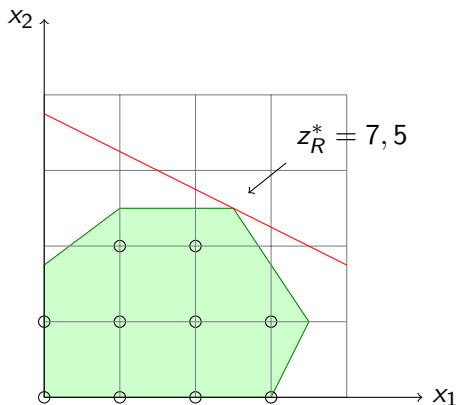
$$-3x_1 + 4x_2 \leq 7$$

$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1, x_2 \in \mathbb{N}$$



Coupes d'arrondi

Ajout d'une contrainte :

$$\max z = x_1 + 2x_2$$

Sous

$$-3x_1 + 4x_2 \leq 7$$

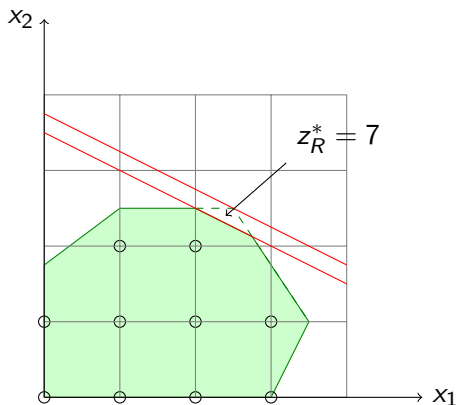
$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1 + 2x_2 \leq [7, 5]$$

$$x_1, x_2 \in \mathbb{N}$$



Coupes d'arrondi

Problème (P) :

$$\max z = x_1 + 2x_2$$

Sous

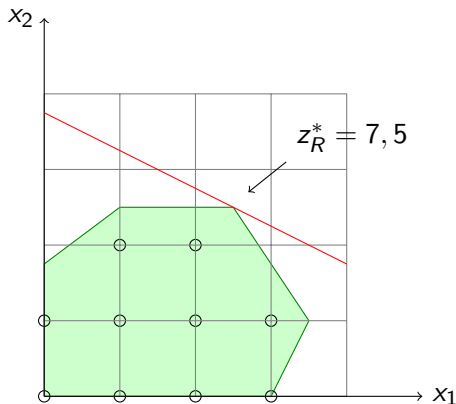
$$-3x_1 + 4x_2 \leq 7$$

$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1, x_2 \in \mathbb{N}$$



Coupes d'arrondi

Contrainte (2) :

$$\max z = x_1 + 2x_2$$

Sous

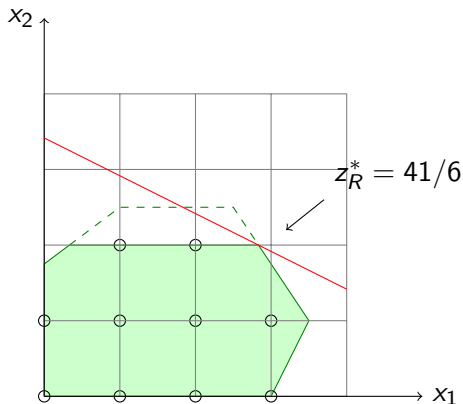
$$-3x_1 + 4x_2 \leq 7$$

$$x_2 \leq \lfloor 5/2 \rfloor$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1, x_2 \in \mathbb{N}$$



Coupes d'arrondi

Fonction objectif :

$$\max z = x_1 + 2x_2$$

Sous

$$-3x_1 + 4x_2 \leq 7$$

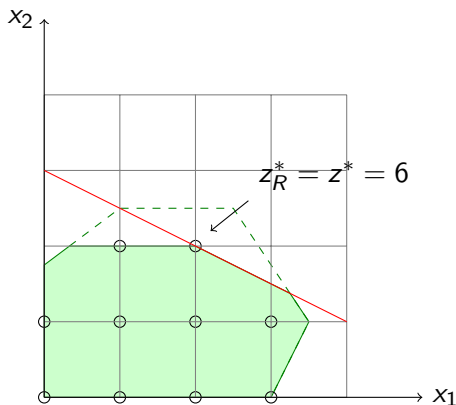
$$x_2 \leq 2$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

$$x_1 + 2x_2 \leq \lfloor 41/6 \rfloor$$

$$x_1, x_2 \in \mathbb{N}$$

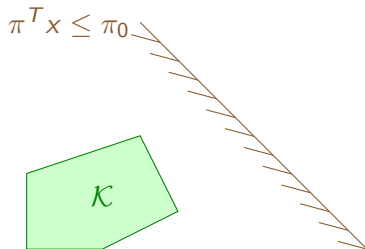


Définitions

Soit $\mathcal{K} = \{x \in \mathbb{R}^n : Ax \leq b\}$ un polyèdre de $\mathbb{R}^n$.

- Une inégalité $\pi^T x \leq \pi_0$ est dite **valide** pour $\mathcal{K}$ si elle est satisfaite pour tout x appartenant à $\mathcal{K}$:

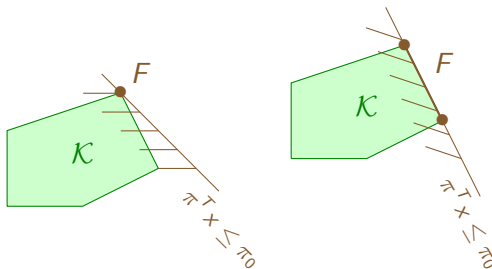
$$\mathcal{K} \subseteq \{x \in \mathbb{R}^n : \pi^T x \leq \pi_0\}$$



Définitions

Facettes d'un polyèdre

- Soit $\pi^T x \leq \pi_0$ une inégalité valide pour le polyèdre $\mathcal{K}$ de $\mathbb{R}^n$.
- $F = \{x \in \mathcal{K} : \pi^T x = \pi_0\}$ est une **face** de $\mathcal{K}$ si $F \neq \emptyset$.
- $F = \{x \in \mathcal{K} : \pi^T x = \pi_0\}$ est une **facette** de $\mathcal{K}$ si $F \neq \emptyset$ et F est de dimension $n - 1$.



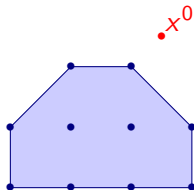
Définitions

Hyperplan de séparation

- Soit $S = \{x^1, x^2, \dots, x^k\}$ ensemble de points de $\mathbb{R}^n$.
- Soit $x^0 \in \mathbb{R}^n \setminus \text{CONV}(S)$.
- Alors il existe $\pi \in \mathbb{R}^n$ et $\pi_0 \in \mathbb{R}$ tels que

$$\pi^T x \leq \pi_0 \quad \forall x \in \text{CONV}(S)$$

$$\pi^T x^0 > \pi_0$$



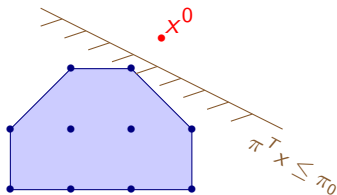
Définitions

Hyperplan de séparation

- Soit $S = \{x^1, x^2, \dots, x^k\}$ ensemble de points de $\mathbb{R}^n$.
- Soit $x^0 \in \mathbb{R}^n \setminus \text{CONV}(S)$.
- Alors il existe $\pi \in \mathbb{R}^n$ et $\pi_0 \in \mathbb{R}$ tels que

$$\pi^T x \leq \pi_0 \quad \forall x \in \text{CONV}(S)$$

$$\pi^T x^0 > \pi_0$$



Ajout de coupes en PLNE

Détermination d'hyperplans de séparation entre

- La solution de la relaxation linéaire du problème, si elle n'est pas entière
- L'enveloppe convexe des solutions entières du problème.

L'enjeu est de trouver les coupes les plus performantes : celles qui permettent de se rapprocher le plus vite de la solution optimale du problème en nombres entiers.

Coupes spécifiques

Coupes adaptées à une catégorie de problèmes

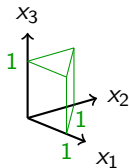
- Coupes de cliques
- Coupes de couverture (sac-à-dos)
- *etc.*

Coupes de cliques

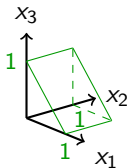
- Clique : ensemble de variables binaires $C = \{x_1, \dots, x_n\}$ tel qu'il existe une contrainte d'exclusion entre chaque couple de variables de C :

$$x_{j1} + x_{j2} \leq 1 \quad \forall (x_{j1}, x_{j2}) \in C^2, j1 < j2$$

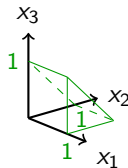
Exemple à trois variables x_1, x_2, x_3



$$x_1 + x_2 \leq 1$$



$$x_1 + x_3 \leq 1$$

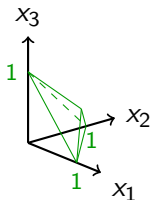


$$x_2 + x_3 \leq 1$$

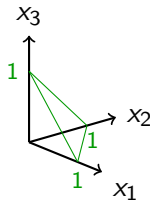
Coupes de cliques

Pour toute clique $C = \{x_1, \dots, x_n\}$ l'inégalité suivante est valide :

$$\sum_{j=1, \dots, n} x_j \leq 1$$



Trois contraintes d'exclusion



Coupe de clique $x_1 + x_2 + x_3 \leq 1$

Coupes de couverture

On considère le problème de sac à dos :

$$\max c^T x$$

sous les contraintes :

$$a \cdot x \leq b$$

$$x \in \{0; 1\}^n$$

$$\text{où } c \in \mathbb{Z}_+^n, a^T \in \mathbb{Z}_+^n, b \in \mathbb{Z}_+.$$

Coupes de couverture

- Soit $C \subset J = \{1, \dots, n\}$. On dit que C est un **ensemble dépendant** si $\sum_{j \in C} a_j > b$.
- Si C est un ensemble dépendant, alors l'inégalité :

$$\sum_{j \in C} x_j \leq |C| - 1,$$

appelée coupe de couverture, est valide.

- Sous certaines conditions, les coupes de couverture peuvent correspondre à des facettes du problème du sac à dos.

Génération de coupes

Principe :

- Coupes issues de la structure du programme linéaire
- Excluent la solution courante
- Sont valides pour l'ensemble des solutions entières

Génération de coupes

Inégalités de Chvátal-Gomory

- Problème (P) : $\max z = c^T x$ sous $Ax \leq b$ et $x \in \mathbb{N}^n$
- Toutes les lignes $A_i \cdot x \leq b_i \quad i = 1, \dots, m$ sont des inégalités valides pour l'enveloppe convexe des solutions admissibles de (P) .
- Soit $y \in \mathbb{R}_+^m$
- Alors $\pi^T x \leq \pi_0$, avec

$$\pi^T = \left[\sum_{i=1}^m y_i \cdot A_i \right] \in \mathbb{Z}^n$$
$$\pi_0 = \left[\sum_{i=1}^m y_i \cdot b_i \right] \in \mathbb{Z}$$

est une inégalité valide pour l'enveloppe convexe des solutions entières de (P) .

Génération de coupes

Propriété

Toute inégalité valide pour l'enveloppe convexe des solutions admissibles de P peut être obtenue en générant un nombre fini d'inégalités de Chvátal-Gomory.

Coupes fractionnaires de Gomory

Problème mis sous forme simpliciale dans la base B :

$$x_B = B^{-1}b - B^{-1}N.x_N \quad z = c_B^T B^{-1}b + \sum_{j \in N} \bar{c}_j x_j$$

Considérons la i^e contrainte :

- $x_{B_i} = (B^{-1}b)_i - \sum_{j \in N} \bar{a}_{ij} x_j$
- Si $(B^{-1}b)_i$ n'est pas entier, on introduit une coupe.
- Soit f_i la partie fractionnaire de $(B^{-1}b)_i$ et f_{ij} la partie fractionnaire de $\bar{a}_{ij}$.

$\sum_{j \in N} f_{ij} x_j \geq f_i$ est une coupe associée à la contrainte i .

Remarque : Il n'est pas obligatoire d'atteindre la base optimale de la relaxation.

Exemple

Problème (P) :

$$\max z = x_1 + 2x_2$$

Sous

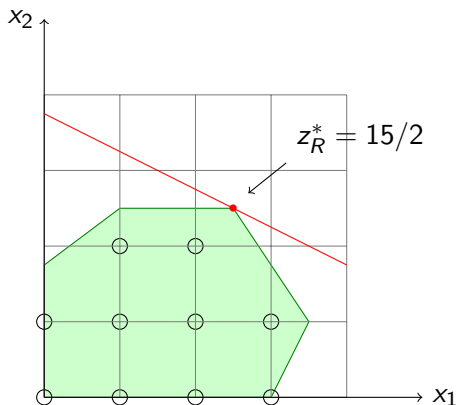
$$-3x_1 + 4x_2 \leq 7$$

$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

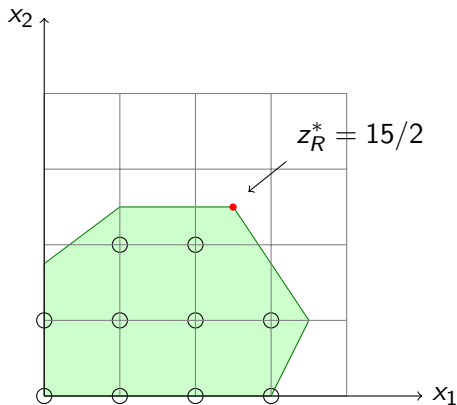
$$x_1, x_2 \in \mathbb{N}$$



Exemple

Tableau final de (R) :

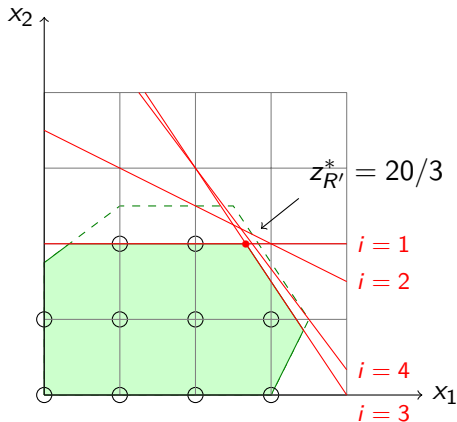
		x_4	x_5
x_1	$5/2$	$-1/2$	0
x_2	$5/2$	$1/3$	$-1/6$
x_3	$9/2$	3	$-1/2$
x_6	$7/2$	$-7/6$	$1/3$
z_R	$15/2$	$-7/6$	$-1/6$



Exemple

Tableau final de (R) :

		x_4	x_5
x_1	$5/2$	$-1/2$	0
x_2	$5/2$	$1/3$	$-1/6$
x_3	$9/2$	3	$-1/2$
x_6	$7/2$	$-7/6$	$1/3$
z_R	$15/2$	$-7/6$	$-1/6$



Plan du cours

1 Introduction

2 Résolution

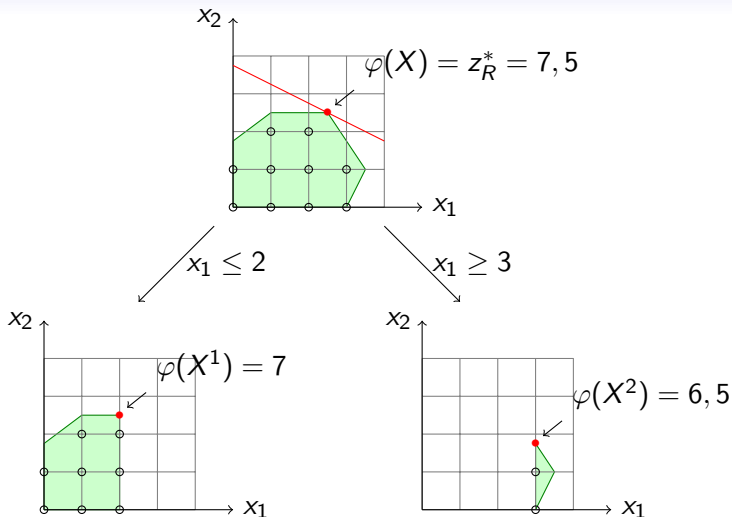
3 Coupes

4 **Recherche Arborescente**

5 Solveurs

6 Modélisation

Recherche Arborescente



Méthodes par séparation et évaluation

Cas général : maximisation de $f(x)$ avec $x \in X$.

- **Séparation** de l'ensemble des solutions admissibles X en plusieurs sous-ensembles.
- Dans chaque sous-ensemble l'**évaluation** de f donne une borne supérieure de l'optimum sur ce sous-ensemble.
- **Comparaison** des résultats sur les différents sous-ensembles pour ne pas développer toute l'arborescence.

Algorithme

- Parcours de l'arborescence jusqu'à obtention de l'optimum ou conclusion sur son existence.

Séparation

- Le premier ensemble à séparer est X
- X est séparé en $X^1, X^2, \dots, X^K$ tels que
 - On ne perd pas de solution
 $X^1 \cup X^2 \cup \dots \cup X^K = X$
 - On ne crée pas de solution
 $X^k \subset X \quad k = 1, \dots, K$

Questions : comment séparer ? Combien de sous-ensembles ?

Évaluation

On sait évaluer un sous-ensemble X^i si on connaît une relaxation du problème :

- L'évaluation $\varphi(X^i)$ est telle que $\varphi(X^i) \geq f(x) \quad \forall x \in X^i$
- L'évaluation est **exacte** si on connaît $\hat{x}^i \in X^i$ admissible tel que $\varphi(X^i) = f(\hat{x}^i)$
- L'évaluation φ' est meilleure que φ si $\varphi'(X^i) \leq \varphi(X^i)$
- Une solution admissible de X^i donne une borne inférieure sur ce sous-ensemble et tous ses parents

Questions : quelle(s) relaxation(s) choisir ?

Sondage

On sonde les sous-ensembles pour conclure sur la présence de l'optimum global :

- **Infaisabilité** : s'il n'y a plus de solution admissible dans le sous-ensemble
- **Optimalité** : si on a trouvé une évaluation exacte du sous-ensemble
- **Dominance** : si on connaît une solution admissible x^k telle que

$$\varphi(X^i) \leq f(x^k)$$

Si on peut conclure sur un sous-ensemble, il n'est plus décomposé (**élagage**).

Stratégies de recherche

Parcours de l'arborescence :

- En largeur : choix du plus ancien nœud apparu.
- En profondeur : choix du dernier nœud apparu.
- Meilleur d'abord : choix du nœud le plus prometteur suivant un critère donné.
- Stratégies hybrides.

Application à la PLNE

Variables entières ou binaires

- Évaluation : relaxation linéaire
- Séparation : ajout de contraintes, fixation d'une variable binaire
- Parcours de l'arborescence
- Association avec des coupes : Branch and Cut

Plan du cours

1 Introduction

2 Résolution

3 Coupes

4 Recherche Arborescente

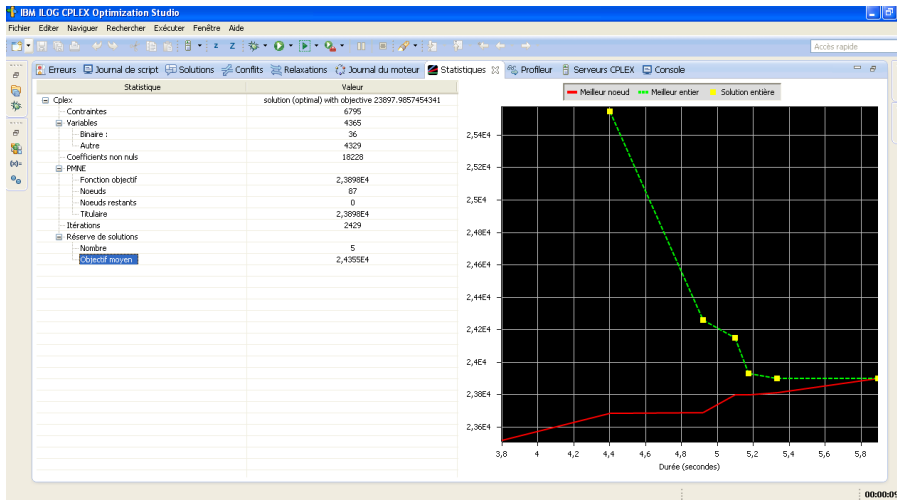
5 **Solveurs**

6 Modélisation

Briques de base des solveurs

- Prétraitement
- Ajout de coupes
- Stratégies de branchement
- Recherche de solutions admissibles

Solveurs : CPLEX



Solveurs : CPLEX

	Nodes					Cuts/		
	Node	Left	Objective	IInf	Best Integer	Best Bound	ItCnt	Gap
	0	0	23517.5108	14		23517.5108	1377	
	0	0	23519.1653	21		Cuts: 18	1425	
	0	0	23684.5035	29		Cuts: 4	1441	
*	0+	0			25542.1080	23684.5035	1441	7.27%
	0	0	23689.1050	29	25542.1080	Flowcuts: 1	1442	7.25%
*	0+	0			24259.8755	23689.1050	1442	2.35%
	0	2	23689.1050	29	24259.8755	23717.3845	1442	2.24%
Elapsed time = 1.86 sec. (369.40 ticks, tree = 0.00 MB, solutions = 2)								
*	10+	10			24147.3112	23797.9064	1794	1.45%
*	11	5	integral	0	23926.6256	23797.9064	1814	0.54%
*	18	5	integral	0	23897.9857	23810.3168	1929	0.37%

Cover cuts applied: 1
 Implied bound cuts applied: 1
 Flow cuts applied: 7
 Mixed integer rounding cuts applied: 2
 Lift and project cuts applied: 1

Root node processing (before b&c):
 Real time = 1.87 sec. (369.51 ticks)
 Sequential b&c:
 Real time = 0.87 sec. (171.75 ticks)

 Total (root+branch&cut) = 2.74 sec. (541.26 ticks)

A retenir

Complexité de la recherche arborescente

- Par définition, il existe dans l'arborescence un chemin polynomial jusqu'à la solution optimale. En revanche, le chemin suivi dans le pire cas est **exponentiel**.
- Les algorithmes de séparation et évaluation (*branch and bound*) explorent l'arborescence de manière **heuristique**. Cette recherche peut être malchanceuse !

Leviers :

- Modélisation : trouver des formulations performantes (qualité de la relaxation / compacité)
- Résolution : définir les bonnes stratégies
- Autres : solutions de départ, heuristiques, décomposition, etc.

Plan du cours

1 Introduction

2 Résolution

3 Coupes

4 Recherche Arborescente

5 Solveurs

6 Modélisation

Modélisation

Objectif :

Élaborer des modèles pertinents et dont la résolution est la plus performante possible.

Deux dimensions à équilibrer :

- Qualité de la relaxation linéaire
- Nombre de variables et de contraintes : compacité du modèle

Modélisation

Dans cette partie du cours : modélisation de non-linéarités

- Problèmes d'ordonnancement : disjonctions et contraintes "Big-M"
- Problèmes de localisation d'entrepôt : agrégation de contraintes
- Modélisation de non linéarités : maximum, valeur absolue, multiplication de variables
- Modélisation de fonctions logiques

Formulation en PLNE d'un problème d'optimisation

Soit un problème d'optimisation général (PO) comportant $p \geq 0$ variables entières et $q \geq 0$ variables continues :

(PO) Maximiser $f(x)$ sous les contraintes $x \in \mathcal{X} \subset \mathbb{N}^p \times \mathbb{R}^q$

La fonction objectif f et les contraintes définissant $\mathcal{X}$ ne sont pas nécessairement linéaires.

La modélisation consiste à trouver une bonne formulation PLNE (ou PL) de (PO) .

Formulation en PLNE d'un problème d'optimisation

Dans le cas le plus général, une reformulation linéaire (P') de (PO) s'écrit en ajoutant des variables entières (ou binaires) et/ou des variables continues nécessaires à la linéarisation.

On note :

- $\mathcal{X}'$ l'ensemble des solutions admissibles de (P')
- $proj_x(\mathcal{X}')$ la projection de $\mathcal{X}'$ sur l'espace des variables originales

La formulation est correcte si

- $proj_x(\mathcal{X}') = \mathcal{X}$
- et si pour tout $x \in \mathcal{X}$, $f(x) = c^T x + c'^T y^*$ où y^* est la solution optimale du PLVM en ayant fixé les variables x .

Exemple

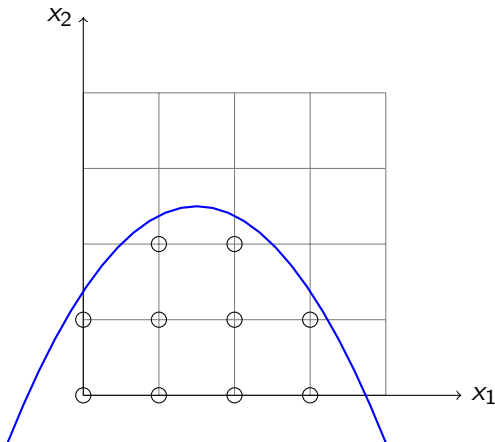
Problème (PO) à résoudre :

$$\max z = x_1 + 2x_2$$

Sous

$$x_2 \leq \frac{5 - (x_1 - 3/2)^2}{2}$$

$$x_1, x_2 \in \mathbb{N}$$



Exemple

(P') formulation correcte
de (PO) :

$$\max z_1 = x_1 + 2x_2$$

Sous

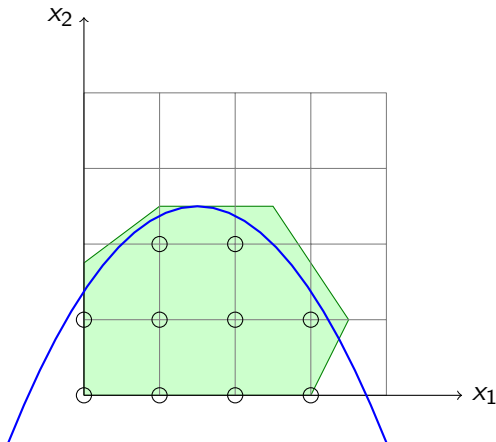
$$-3x_1 + 4x_2 \leq 7$$

$$2x_2 \leq 5$$

$$6x_1 + 4x_2 \leq 25$$

$$2x_1 - x_2 \leq 6$$

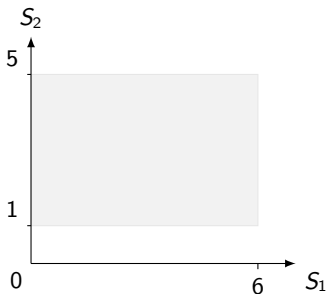
$$x_1, x_2 \in \mathbb{N}$$



Modélisation d'une disjonction

Exemple

Un simple problème d'ordonnancement à deux tâches J_1 et J_2 de durées $p_1 = 3$ et $p_2 = 2$ sur une machine ne pouvant exécuter qu'une tâche à la fois. On doit déterminer les dates de débuts continues S_1 et S_2 sachant que chaque tâche doit démarrer dans une fenêtre de temps ($[0, 6]$ pour J_1 et $[1, 5]$ pour J_2). On cherche à minimiser $f(S) = S_1 + S_2 + p_1 + p_2$.



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

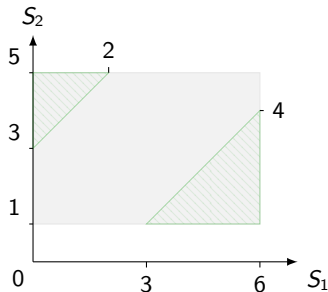
$$1 \leq S_2 \leq 5$$

$$S_2 \geq S_1 + 3 \vee S_1 \geq S_2 + 2$$

Modélisation d'une disjonction

Exemple

Un simple problème d'ordonnancement à deux tâches J_1 et J_2 de durées $p_1 = 3$ et $p_2 = 2$ sur une machine ne pouvant exécuter qu'une tâche à la fois. On doit déterminer les dates de débuts continues S_1 et S_2 sachant que chaque tâche doit démarrer dans une fenêtre de temps ($[0, 6]$ pour J_1 et $[1, 5]$ pour J_2). On cherche à minimiser $f(S) = S_1 + S_2 + p_1 + p_2$.



L'ensemble des solutions admissibles $\mathcal{X}$ est disjoint.

(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

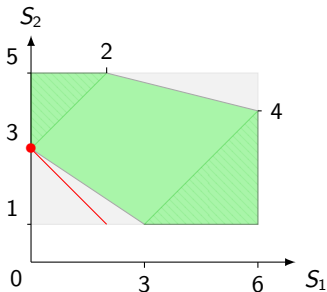
$$1 \leq S_2 \leq 5$$

$$S_2 \geq S_1 + 3 \vee S_1 \geq S_2 + 2$$

Modélisation d'une disjonction

Exemple

Un simple problème d'ordonnancement à deux tâches J_1 et J_2 de durées $p_1 = 3$ et $p_2 = 2$ sur une machine ne pouvant exécuter qu'une tâche à la fois. On doit déterminer les dates de débuts continues S_1 et S_2 sachant que chaque tâche doit démarrer dans une fenêtre de temps ($[0, 6]$ pour J_1 et $[1, 5]$ pour J_2). On cherche à minimiser $f(S) = S_1 + S_2 + p_1 + p_2$.



(PO) peut être résolu par la PL sur $\text{conv}(\mathcal{X})$.

Impraticable en général

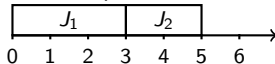
(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

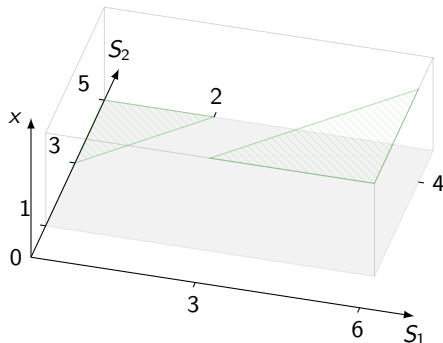
$$S_2 \geq S_1 + 3 \vee S_1 \geq S_2 + 2$$

Solution optimale de valeur 8



Modélisation d'une disjonction

On introduit une variable binaire x pour représenter les deux alternatives.



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

$$S_2 \geq S_1 + 3 - M.x$$

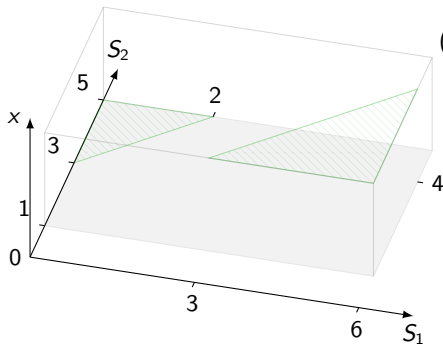
$$S_1 \geq S_2 + 2 - M'(1 - x)$$

$$x \in \{0, 1\}$$

Ajout de contraintes "Big-M".

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

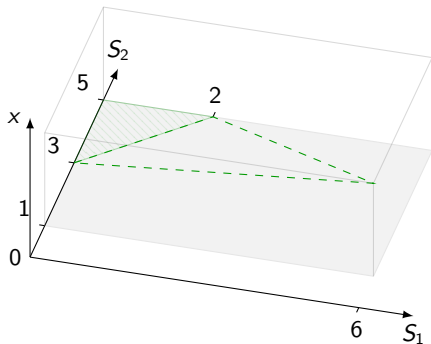
$$S_2 \geq S_1 + 3 - M \cdot x$$

$$S_1 \geq S_2 + 2 - M'(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

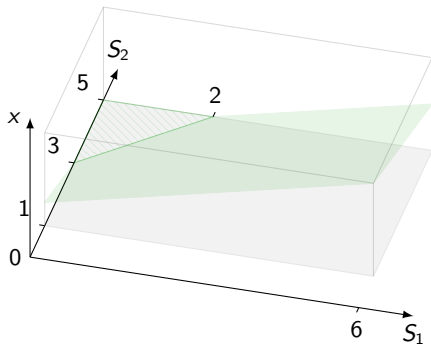
$$S_2 \geq S_1 + 3 - M.x$$

$$S_1 \geq S_2 + 2 - M'.(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

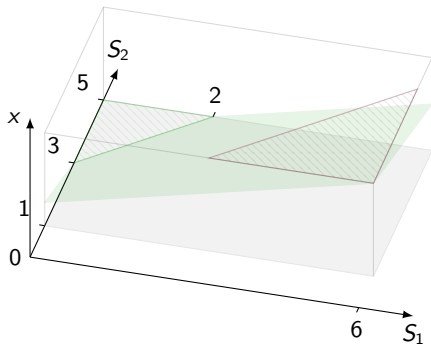
$$S_2 \geq S_1 + 3 - 8.x$$

$$S_1 \geq S_2 + 2 - M'(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

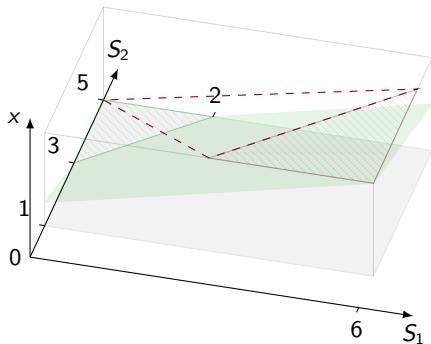
$$S_2 \geq S_1 + 3 - 8.x$$

$$S_1 \geq S_2 + 2 - M'(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

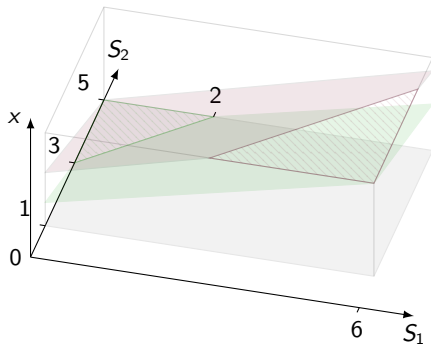
$$S_2 \geq S_1 + 3 - 8x$$

$$S_1 \geq S_2 + 2 - M'(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

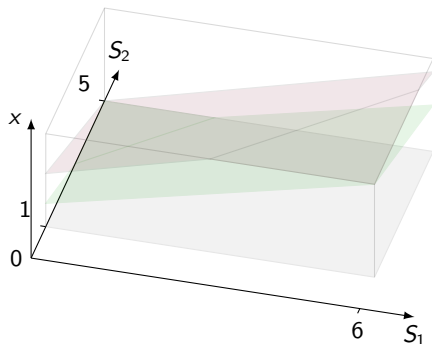
$$S_2 \geq S_1 + 3 - 8x$$

$$S_1 \geq S_2 + 2 - 7(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

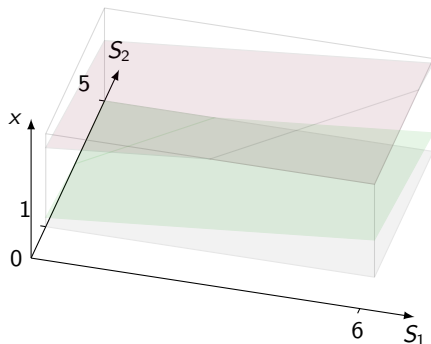
$$S_2 \geq S_1 + 3 - 8.x$$

$$S_1 \geq S_2 + 2 - 7(1 - x)$$

$$x \in \{0, 1\}$$

Modélisation d'une disjonction

Qualité de la relaxation : ajustement des valeurs de M et M' .



(PO) Minimiser $S_1 + S_2 + 5$

$$0 \leq S_1 \leq 6$$

$$1 \leq S_2 \leq 5$$

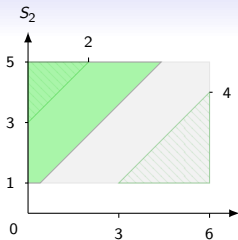
$$S_2 \geq S_1 + 3 - 20 \cdot x$$

$$S_1 \geq S_2 + 2 - 20(1 - x)$$

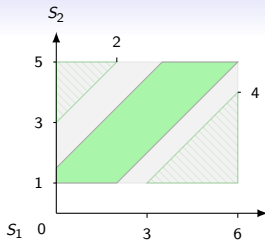
$$x \in \{0, 1\}$$

Mauvaise relaxation

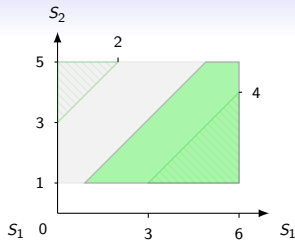
Modélisation d'une disjonction



$x = 0.3$



$x = 0.5$



$x = 0.7$

Qualité des relaxations : formulations agrégées et désagrégées

Problème de localisation d'entrepôt

Une entreprise livre des produits à m clients et peut construire pour cela n entrepôts potentiels. Ouvrir un entrepôt j entraîne un coût fixe f_j et le transport d'une commande de l'entrepôt i vers le client j a également un coût c_{ij} de telle sorte que si on transporte une fraction de la commande d'un client j depuis l'entrepôt i , le coût est de c_{ij} fois cette fraction. Quels entrepôts doivent être ouverts et quel est le plan de transport pour couvrir 100% des commandes à cout minimal ?

- Ouverture d'un entrepôt : variable binaire y_i
- Pourcentage de la commande du client i transportée depuis l'entrepôt j : variable continue $x_{ij} \in [0, 1]$.

Qualité des relaxations : formulations agrégées et désagrégées

Minimiser $\sum_{i=1}^n \sum_{j=1}^m c_{ij} x_{ij} + \sum_{i=1}^n p_i y_i$

Sous les contraintes

$$\left\{ \begin{array}{ll} \sum_{i=1}^n x_{ij} = 1 & \forall j = 1, \dots, m \\ \sum_{j=1}^m x_{ij} \leq m y_i & \forall i = 1, \dots, n \\ y_i \in \{0, 1\} & \forall i = 1, \dots, n \\ x_{ij} \geq 0 & \forall i = 1, \dots, n; \forall j = 1, \dots, m \end{array} \right.$$

La contrainte $\sum_{j=1}^m x_{ij} \leq m y_i \quad \forall i = 1, \dots, n$ (1) modélise l'implication

$$y_i = 0 \implies \sum_{j=1}^m x_{ij} = 0$$

Qualité des relaxations : formulations agrégées et désagrégées

Une alternative est de modéliser m implications

$$y_i = 0 \implies x_{ij} = 0, \forall j = 1, \dots, m$$

On obtient des contraintes désagrégées

$$x_{ij} \leq y_i, \forall i = 1, \dots, n, \forall j = 1, \dots, m \quad (2)$$

Théorème

La formulation désagrégée est strictement meilleure que la formulation agrégée.

Preuve. Soit P_1 le polyèdre correspondant aux contraintes agrégées (1) et soit P_2 le polyèdre remplaçant les contraintes (1) par les contraintes désagrégées (2). Comme (2) $\implies$ (1), on a $P_2 \subseteq P_1$. Pour prouver l'inclusion stricte il suffit de trouver un point $x \in P_1 \setminus P_2$. Supposons que $m = kn$. Soit le point x tel que $x_{ij} = 1$ pour $i = 1, \dots, n; j = k(i-1) + 1, \dots, k(i-1) + k$; $x_{ij} = 0$ sinon et $y_i = k/m$ pour $i = 1, \dots, n$. x ainsi défini satisfait (1) mais viole (2). $\square$

Modélisation de non linéarités

Possibilité de linéariser des expressions qu'on rencontre fréquemment :

- Min, Max, valeur absolue
- Fonctions linéaires par morceaux
- Multiplication de variables
- Expressions logiques

Modélisation d'un minimum

- $f : [0, C]^2 \mapsto [0, C], \quad z = f(x, y) = \text{Min}(x, y)$

$$\left\{ \begin{array}{l} z \leq x \\ z \leq y \\ 0 \leq x \leq C \\ 0 \leq y \leq C \end{array} \right.$$

Insuffisant

Modélisation d'un minimum

• $f : [0, C]^2 \mapsto [0, C], \quad z = f(x, y) = \text{Min}(x, y)$

$$\left\{ \begin{array}{l} z \leq x \\ z \leq y \\ z \geq x - Cw \\ z \geq y - C(1 - w) \\ w \in \{0, 1\} \\ 0 \leq x \leq C \\ 0 \leq y \leq C \end{array} \right.$$

Modélisation d'un minimum

• $f : [0, C]^2 \mapsto [0, C], \quad z = f(x, y) = \text{Min}(x, y)$

La modélisation de f peut être simplifiée si le coefficient de z dans l'objectif est positif, dans le cadre d'une maximisation

Maximiser az

Sous les contraintes

$$\begin{cases} z \leq x \\ z \leq y \\ 0 \leq x \leq C \\ 0 \leq y \leq C \end{cases}$$

avec $a > 0$

$$\begin{cases} z \leq x \\ z \leq y \\ z \geq x - Cw \\ z \geq y - C(1 - w) \\ w \in \{0, 1\} \\ 0 \leq x \leq C \\ 0 \leq y \leq C \end{cases}$$

Modélisation d'une valeur absolue

- $f : [a, b] \mapsto \mathbb{R}, z = f(x) = |x|$ avec $a < 0 < b$

$$\left\{ \begin{array}{l} z = x^+ + x^- \\ x = x^+ - x^- \\ a \leq x \leq b \\ 0 \leq x^+ \leq b.y \\ 0 \leq x^- \leq -a(1 - y) \\ y \in \{0, 1\} \end{array} \right.$$

Modélisation d'une valeur absolue

- $f : [a, b] \mapsto \mathbb{R}, z = f(x) = |x|$ avec $a < 0 < b$

$$\begin{cases} z = x^+ + x^- \\ x = x^+ - x^- \\ a \leq x \leq b \\ 0 \leq x^+ \leq b \cdot y \\ 0 \leq x^- \leq -a(1 - y) \\ y \in \{0, 1\} \end{cases}$$

Simplification si l'objectif minimise z

Min z

$$\begin{cases} z = x^+ + x^- \\ x = x^+ - x^- \\ a \leq x \leq b \\ 0 \leq x^+ \leq b \\ 0 \leq x^- \leq -a \end{cases}$$

Multiplication de variables binaires

Produit de deux variables binaires x_i et x_j : $z = x_i \cdot x_j$

$$z \leq x_i$$

$$z \leq x_j$$

$$z \geq x_i + x_j - 1$$

Généralisation (Watters, 1967) : le produit de variables binaires $x_j, j \in J$ peut être remplacé par $z \in \{0, 1\}$ tel que :

$$\sum_{j \in J} x_j - z \leq |J| - 1$$
$$-\sum_{j \in J} x_j + |J| \cdot z \leq 0$$

Multiplication de variables mixtes

Produit d'une variable binaire x et d'une variable réelle positive bornée $y \in [0; y_{sup}]$: $z=x.y$

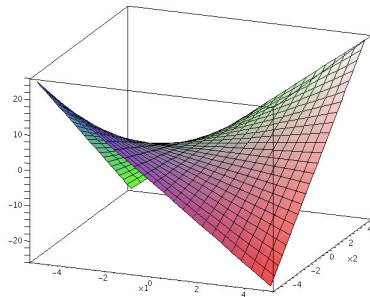
$$z \leq y$$

$$z \leq x.y_{sup}$$

$$z \geq y - y_{sup}(1 - x)$$

Multiplication de deux variables continues

Contrainte bilinéaire : $z = x.y$



Multiplication de deux variables continues

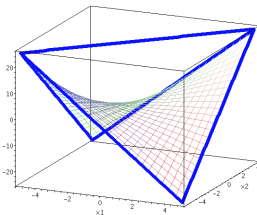
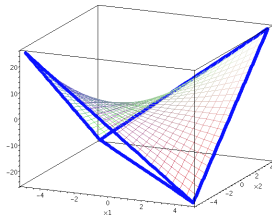
Relaxation de Mc Cormick (1976)

$$z \geq x_{inf} \cdot y + y_{inf} \cdot x - x_{inf} \cdot y_{inf}$$

$$z \geq x_{sup} \cdot y + y_{sup} \cdot x - x_{sup} \cdot y_{sup}$$

$$z \leq x_{sup} \cdot y + y_{inf} \cdot x - x_{sup} \cdot y_{inf}$$

$$z \leq x_{inf} \cdot y + y_{sup} \cdot x - x_{inf} \cdot y_{sup}$$



Modélisation de fonctions logiques

Rappels d'algèbre booléenne

- Une **proposition** est un énoncé qui est soit vrai, soit faux mais pas les deux à la fois (« tiers exclu »).
- La valeur de vérité de la proposition est dite **vraie** (**V**) si la proposition est vérifiée ; elle est **fausse** (**F**) dans le cas contraire.
- Soit P une proposition. La **négation** de P , notée $\neg P$, est un opérateur logique qui prend la valeur opposée de P .

Rappels d'algèbre booléenne

On appelle **connecteur logique** un opérateur qui permet de former une nouvelle proposition à partir de deux propositions ou plus.

- **Conjonction** : la proposition « A et B », notée $A \wedge B$, est vraie si et seulement si A et B sont vraies.
- **Disjonction** : la proposition « A ou B », notée $A \vee B$, est fausse si et seulement si A et B sont fausses.
- **Implication** : la proposition « A implique B », notée $A \rightarrow B$, est fausse si et seulement si A est vraie et B est fausse.
- **Equivalence** : notée $A \leftrightarrow B$, est vraie si et seulement si A et B ont la même valeur de vérité.

Rappels d'algèbre booléenne

Équivalence entre les propositions

Nom	Équivalence
Identité	$A \wedge \mathbf{V} \equiv A$ $A \vee \mathbf{F} \equiv A$
Domination	$A \vee \mathbf{V} \equiv \mathbf{V}$ $A \wedge \mathbf{F} \equiv \mathbf{F}$
Idempotence	$A \wedge A \equiv A$ $A \vee A \equiv A$
Double négation	$\neg(\neg A) \equiv A$
Commutativité	$A \wedge B \equiv B \wedge A$ $A \vee B \equiv B \vee A$
Associativité	$(A \wedge B) \wedge C \equiv A \wedge (B \wedge C)$ $(A \vee B) \vee C \equiv A \vee (B \vee C)$

Rappels d'algèbre booléenne

Équivalence entre les propositions

Nom	Équivalence
Distributivité	$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$ $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
Loi de De Morgan	$\neg(A \wedge B) \equiv \neg A \vee \neg B$ $\neg(A \vee B) \equiv \neg A \wedge \neg B$
Autres	$A \vee \neg A \equiv \mathbf{V}$ $A \wedge \neg A \equiv \mathbf{F}$ $A \rightarrow B \equiv \neg A \vee B$

Traduction en équations linéaires

Soit x_P une variable binaire associée à la proposition P . La variable x_P vaut 1 si la proposition P est vraie, 0 sinon.

Un **littéral** est une variable x_P ou sa négation $\neg x_P$.

Une **clause** est une disjonction littéraux. Ex. $\neg x_{P1} \vee x_{P2} \vee x_{P3}$

Pour traduire une expression logique en équations linéaires, on doit l'exprimer sous la forme d'une clause ou d'une conjonction de clauses.

Traduction en équations linéaires

Expression	Contraintes linéaires
A vrai	$x_A \geq 1$
$\neg A$ vrai	$1 - x_A \geq 1$
$A \wedge B$ vrai	$x_A \geq 1$ $x_B \geq 1$
$A \vee B$ vrai	$x_A + x_B \geq 1$
$A \rightarrow B$ vrai	$x_A \leq x_B$

Traduction en équations linéaires

Expression	Contraintes linéaires
$P \leftrightarrow A$	$x_P = x_A$
$P \leftrightarrow \neg A$	$x_P = 1 - x_A$
$P \leftrightarrow A \wedge B$	$x_P \leq x_A$ $x_P \leq x_B$ $x_P \geq x_A + x_B - 1$
$P \leftrightarrow A \vee B$	$x_P \geq x_A$ $x_P \geq x_B$ $x_P \leq x_A + x_B$
$P \leftrightarrow A \rightarrow B$	$x_P \geq 1 - x_A$ $x_P \geq x_B$ $x_P \leq 1 - x_A + x_B$

Sources et compléments I



Michel Minoux.

Programmation mathématique.

Lavoisier, 2008, 2^e édition.



Dominique de Werra, Thomas Liebling, Jean-François Hêche.

Recherche opérationnelle pour ingénieurs.

Presses polytechniques et universitaires romandes, 2003.



Stephen Bradley, Arnoldo Hax, Thomas Magnanti

Applied mathematical programming.

MIT Press, Addison-Wesley, 1977.

[http ://web.mit.edu/15.053/www/](http://web.mit.edu/15.053/www/)



Laurence Wosley

Integer programming.

Wiley -Interscience, 1998.

Sources et compléments II



François Soumis

Modélisation en recherche opérationnelle.

Support de cours École Polytechnique de Montréal.



Michel Gamache

Recherche opérationnelle minière.

Support de cours École Polytechnique de Montréal.



Andrea Lodi

Problem-solving by Mixed-Integer Programming.

Présentation à ROADEF 2014.