V. Programmation Linéaire

V.1. Introduction

Cyril Briand (briand@laas.fr) – Master 2 AURO

Définition

- Un programme linéaire (PL) est un problème d'optimisation consistant à maximiser (ou minimiser) une fonction objectif linéaire de *n* variables de décision soumises a un ensemble de contraintes exprimées sous forme d'équations ou d'inéquations linéaires.

- À l'origine, le terme programme a le sens de planification opérationnelle mais il est aujourd'hui employé comme synonyme de problème (d'optimisation).

- La terminologie est due a G. B. Dantzig, inventeur de l'algorithme du simplexe (1947).

Hypothèses de la PL

1. **Hypothèse de proportionnalité et additivité ou hypothèse de linéarité**
   La contribution des variables de décision à la fonction objectif et aux contraintes est proportionnelle à leur valeur

   $3x_1$ ✓ $\frac{3}{4} x_2$ ✓ $1/x_1$ ✗ log(x2) ✗

   La contribution de chaque variable est indépendante de la valeur des autres variables

   $z=3x_1+x_2$ ✓ $z=2\, x_1\, x_2$ ✗

   Si elle est violée : **Programmation non linéaire**

2. **Hypothèse de divisibilité**
   Les variables peuvent prendre des valeurs fractionnaires.
   Si elle est violée : **Programmation en nombres entiers**

3. **Hypothèse de déterminisme**
   Chaque paramètre est connu précisément.
   Si elle est violée : **Programmation stochastique**

Forme canonique et standard

- **Tout PL peut être transformé en un PL équivalent**
  - sous forme canonique
  - sous forme standard
- **Equivalent**
  - Sol PL initial ⇔
  - Sol PL canonique ⇔
  - Sol PL standard

$$\underbrace{\begin{array}{lrcl} \text{Max} & z = & \boldsymbol{cx} & \\ \text{s.c.} & \boldsymbol{Ax} & \leq & \boldsymbol{b} \\ & \boldsymbol{x} & \geq & \boldsymbol{0} \end{array}}_{\text{Forme canonique}}$$

$$\underbrace{\begin{array}{lrcl} \text{Max} & z = & \boldsymbol{cx} & \\ \text{s.c.} & \boldsymbol{Ax} & = & \boldsymbol{b} \\ & \boldsymbol{x} & \geq & \boldsymbol{0} \end{array}}_{\text{Forme standard}}$$

- **Passer d'une forme à une autre**
  - "≥" à "≤"

    $\boldsymbol{a.x} \geq b \Leftrightarrow -\boldsymbol{ax} \leq -b$
  - Inégalité à égalité
    - *Variables d'écart s*

    $$\boldsymbol{ax} \leq b \iff \boldsymbol{ax} + s = b,\ s \geq 0$$
  - Égalité à inégalité

    $$\boldsymbol{ax} = b \iff \left\{\begin{array}{lcl} \boldsymbol{ax} & \leq & b \\ \boldsymbol{ax} & \geq & b \end{array}\right. \iff \left\{\begin{array}{rcl} \boldsymbol{ax} & \leq & b \\ (-\boldsymbol{a})\boldsymbol{x} & \leq & -b \end{array}\right.$$
  - x négatif
    - *Variables sup*

    $$x \in \mathbb{R} \iff \left\{\begin{array}{l} x = x^+ - x^- \\ x^+, x^- \geq 0 \end{array}\right.$$
  - Minimisation à maximisation

    $\min \boldsymbol{c.x} = -\max(-\boldsymbol{cx})$

Forme canonique et standard

- **Commentaires**
  - n variables, m contraintes
  - $\boldsymbol{A} \in R^{n+m}$
  - $\boldsymbol{x} \in R^n$, $\boldsymbol{b} \in R^m$
  - $\boldsymbol{c} \in R^n$ *est un vecteur ligne*
  - $m > n$
  - Rang(***A***)=*m*, ie la matrice est de rang plein

$$\underbrace{\begin{array}{llrcl} \text{Max} & z = & \boldsymbol{cx} & & \\ \text{s.c.} & & \boldsymbol{Ax} & = & \boldsymbol{b} \\ & & \boldsymbol{x} & \geq & \boldsymbol{0} \end{array}}_{\text{Forme standard}}$$

- **Remarque**
  - On peut toujours se ramener à Rang(***A***)=*m* car sinon
    - b $\in$ Im(***A***) et le système est ***Ax***=***b*** est redondant (on peut supprimer des contraintes redondantes)
    - b $\notin$ Im(***A***) et le système est ***Ax***=***b*** est *inconsistant*

Terminologie

- **Quelle que soit la forme du PL**
  - $x_1, \ldots x_n$: variables de décisions
  - Si le vecteur x satisfait toutes les contraintes, x est une solution admissible.
  - L'ensemble de toutes les solutions admissibles est l'ensemble admissible ou la région admissible.
  - La fonction $\boldsymbol{cx}$ est la fonction objectif ou fonction de coût.
  - Une solution admissible x^* qui maximise la fonction objectif (i.e. $\boldsymbol{cx^*} \geq \boldsymbol{cx}$ pour tout x admissible) est appelée solution admissible optimale ou solution optimale

V. Programmation Linéaire

V.2. Connexions entre approches géométrique et algébrique

UNIVERSITÉ TOULOUSE III PAUL SABATIER

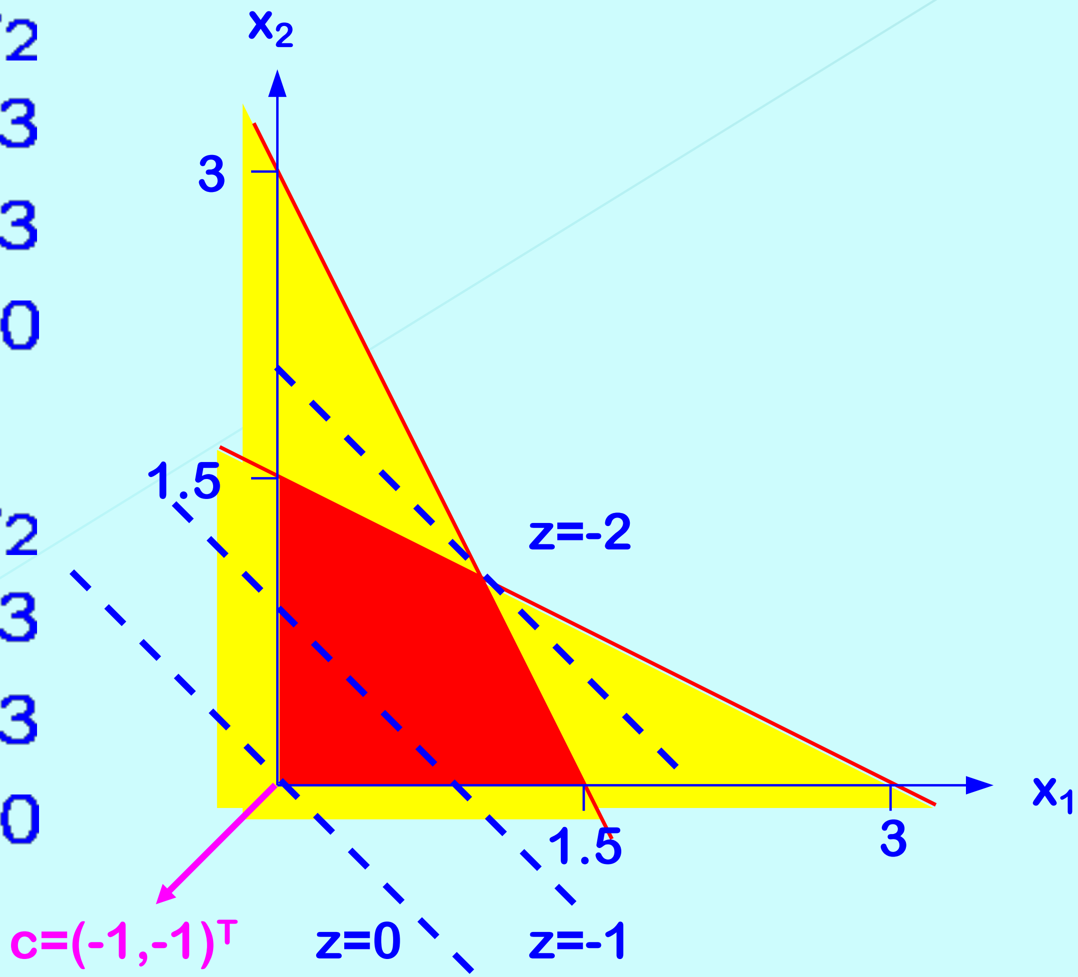
Cyril Briand (briand@laas.fr) – Master 2 AURO

Représentation dans le plan

- **2D => 2 variables de décision**

$$\min z = -x_1 - x_2$$
$$\text{s.c. } x_1 + 2x_2 \leq 3$$
$$2x_1 + x_2 \leq 3$$
$$x_1, x_2 \geq 0$$

$$\min z = -x_1 - x_2$$
$$x_1 + 2x_2 \leq 3$$
$$2x_1 + x_2 \leq 3$$
$$x_1, x_2 \geq 0$$



Premières connexions

- L'ensemble des solutions d'une inéquation (linéaire) correspond a un demi-espace dans R^n (un demi-plan dans R^2)
- L'ensemble des solutions d'une équation (linéaire) correspond a un hyperplan dans R^n (une droite dans R^2)
- L'ensemble des solutions d'un système d'équations et d'inéquations (linéaires) correspond a l'intersection des demi-espaces et des hyperplans associes a chaque élément du système
- Cette intersection, appelée domaine admissible, est convexe et définit un polyèdre dans R^n (une région polygonale dans R^2).
- Le domaine admissible d'un PL peut être :
  - vide: le problème est sans solution admissible (et ne possède évidemment pas de solution optimale).
  - borné (et non vide): le problème possède toujours au moins une solution optimale, quelle que soit la fonction objectif.
  - non borné : alors selon la fonction objectif choisie
    - *il peut exister des solutions admissibles de valeur arbitrairement grande (ou petite) et le PL est dit non borne.*

Algèbre : Solution d'un système d'équations

- Soit $\boldsymbol{Ax} = \boldsymbol{b}$ un système d'équations linéaires avec $\boldsymbol{A} \in \boldsymbol{R}^{n+m}$ et rang($\boldsymbol{A}$) = m

- Pour déterminer les solutions du système
  - **Déterminer *m* colonnes linéairement indépendantes de *A* (formant donc une base *B* de l'espace vectoriel défini par *A*)**
  - **Exprimer le système dans cette base (i.e. multiplié par B^{-1})**

$$Ax = b \overset{P}{\equiv} (B \mid N)\begin{pmatrix} x_B \\ x_N \end{pmatrix} = b \qquad x_B + B^{-1}Nx_N = B^{-1}b$$

- Exemple

$$A = \begin{pmatrix} 0 & \boxed{1} & 2 & \boxed{2} \\ 1 & \boxed{1} & 2 & \boxed{3} \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} \quad \text{et} \quad b = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

$$B = \begin{pmatrix} 1 & 2 \\ 1 & 3 \end{pmatrix} \qquad B^{-1} = \begin{pmatrix} 3 & -2 \\ -1 & 1 \end{pmatrix}.$$

$$B^{-1}(A \mid b) = \left(\begin{array}{cccc|c} -2 & 1 & 2 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \end{array}\right) \Leftrightarrow \begin{cases} x_2 = 1 + 2x_1 - 2x_3 \\ x_4 = -x_1 \end{cases}$$

Algèbre : Base, solution de base

- Toute matrice *B* formée de *m* colonnes linéairement indépendantes de *A* est une **base** du système
- Les variables associées aux colonnes de *B* sont dites "de base" (ou basiques), les autres "hors base"
- La solution particulière obtenue en fixant à 0 les variables hors base est appelée **solution de base** (ou **basique**) associée a la base *B*.

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

- **Solution de base d'un PL standard**

  $$\begin{array}{rcccl} Ax_D & + & Ix_E & = & b \\ x_D & , & x_E & \geq & 0 \end{array}$$

  – **Solution de base évidente *B = I***

  $x_B = x_E = b$ et $x_N = x_D = 0$

  – **Pour une base *B* quelconque**

  - ***Si la $i^{ème}$ variable d'écart (x_{n+i}) est fixée à 0, la solution est sur l'hyperplan***

    $a_{i1}x_1 + \ldots + a_{in}x_n = b_i$ La contrainte est dite **active**

  - ***Si la $j^{ème}$ variable de décision (x_j) est fixée à 0, la solution est sur l'hyperplan***

    $x_j = 0$

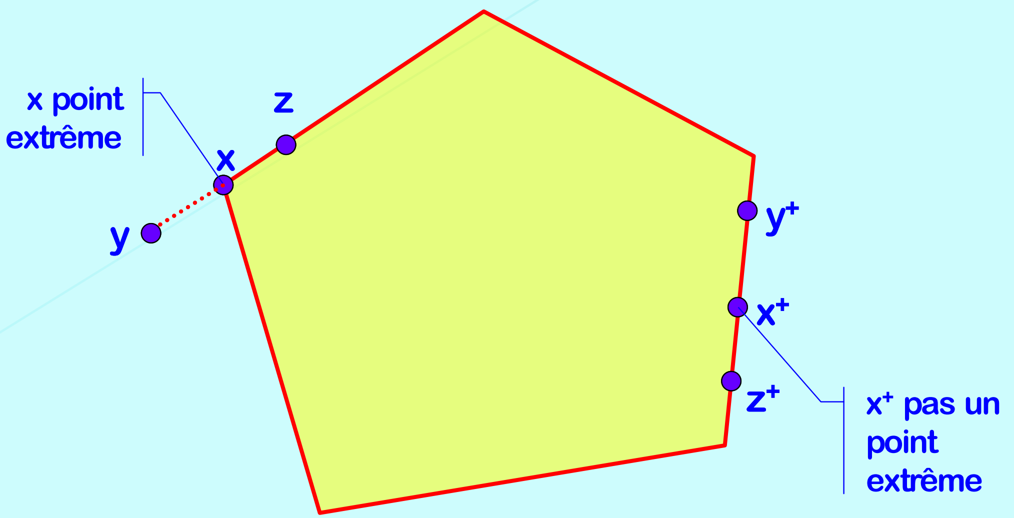
  – La solution de base est **admissible** si $x_B = B^{-1}b$ est ≥ 0

Géométrie : Points extrêmes

Définitions :

- Soit P un polyèdre. Un vecteur $x \in P$ est un point extrême de P si on ne peut pas l'exprimer comme combinaison convexe de deux autres points de P.
- i.e., on ne peut pas trouver deux vecteurs y et z dans P, différents de x, et un scalaire $\lambda \in [0,1]$ tels que

$$x = \lambda\, y + (1-\lambda)\, z$$



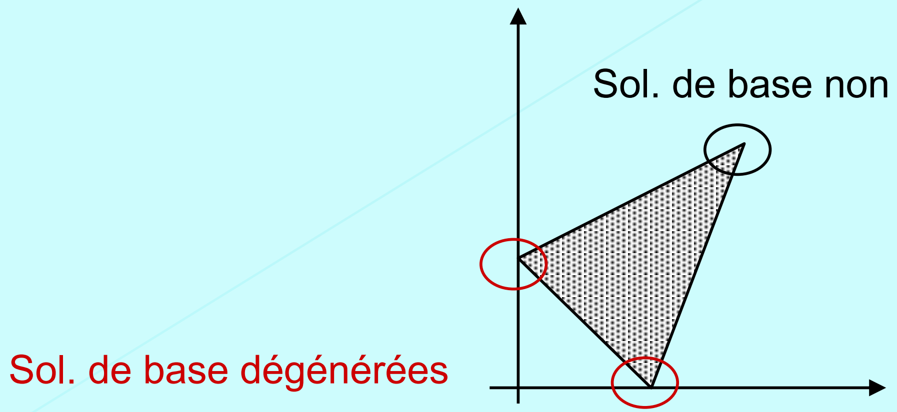
Connexions

- **Concepts géométriques :**
  - Point extrême (ou sommet)
- **Concept algébrique :**
  - Solution de base admissible
- **Soit P un polyèdre, et soit x* ∈ P. Alors,**
  x* est un point extrême ssi
  x* est un sommet ssi
  x* est une solution de base admissible
- **Théorème fondamentale de la PL**
  - Si un PL a une solution admissible, il a une solution de base admissible
  - Si un PL a une solution optimale, il a une solution de base admissible optimale

Dégénérescence

Définitions

- Un PL est dégénéré si, dans l'espace de ses variables de decision, il existe plus de n hyperplans définis pas les contraintes (y compris les contraintes de bornes) se coupant en un seul sommet.
- Une solution de base $x \in IR^n$ est dite dégénérée si au moins une variable de base est nulle (i.e. plus de n contraintes sont actives en x)



V. Programmation Linéaire

V.3. Principe de l'algo du simplexe (phase II)

UNIVERSITÉ TOULOUSE III PAUL SABATIER

Cyril Briand (briand@laas.fr) – Master 2 AURO

Idées de base

- **Méthode du simplexe : passer d'une solution de base admissible à une autre *voisine*, en augmentant la valeur de z, jusqu'à trouver la solution de base optimale**
- **Schéma de l'algorithme:**
  - Soit x_0 une solution de base **admissible**
  - Pour k=0,…. faire
    - *Trouver x_{k+1} sol. de base adm. voisine telle que $c^T x_{k+1} > c^T x_k$*
  - Jusqu'à ce qu'aucune sol. de base adm. voisine n'améliore l'objectif.
- **Il y a au plus** $C_n^m = \dfrac{n!}{m!(n-m)!}$ **solutions de base (n variables, m ctes)**
- **Problème = à chaque itération, comment choisir la prochaine solution de base (voisine) admissible ?**
  - Quelle est la variable hors base *j* à faire rentrer dans la base et quelle est la variable de base à faire sortir
  - **Géométriquement dit** : sur quelle du polyèdre **arête** se déplacer ?

Exemple – Utilisation de dictionnaires

Formulation

- Une fabrique d'objets en terre cuite produit des cendriers, des bols, des cruches et des vases
- La fabrication de chaque objet nécessite 3 phases : moulage, cuisson, peinture ; dont les durées dont indiquées dans le tableau ci-dessous
- La vente de chaque objet rapporte un certain bénéfice
- L'entreprise dispose quotidiennement de 42h de moulage, de 17 heures de cuisson et de 24h de peinture
- Quelle fabrication pour maximiser le chiffre d'affaires (en régime stable) ?

Objet	Cendrier	Bol	Cruche	Vase
Temps de Moulage	2	4	5	7
Temps de Cuisson	1	1	2	2
Temps de peinture	1	2	3	3
Bénéfice	7	9	18	17

MAX Z = 7x1 + 9x2 + 18x3 + 17x4
subject to
2x1 + 4x2 + 5x3 + 7x4 <= 42
x1 + x2 + 2x3 + 2x4 <= 17
x1 + 2x2 + 3x3 + 3x4 <= 24
and x1,x2,x3,x4 >= 0

Tableau du simplexe

- Il y a plusieurs manières d'implémenter l'algorithme du simplexe. La méthode du tableau est l'une des plus classiques.
- Un tableau est une matrice augmentée associée a un PL sous forme standard.

PL standard initial

$$\text{Max} \quad (z, \text{ s.c. } \boldsymbol{x_D}, \boldsymbol{x_E} \geq \mathbf{0})$$

$$\begin{array}{rcrcrcr} \boldsymbol{A x_D} & + & \boldsymbol{I x_E} & & & = & \boldsymbol{b} \\ \hline -\boldsymbol{c_D x_D} & - & \mathbf{0}\boldsymbol{x_E} & + & z & = & 0 \end{array}$$

Tableau initial associé

$$T_0 = \begin{array}{c|c|c|c||c|} \multicolumn{1}{c}{} & \multicolumn{1}{c}{\boldsymbol{x_D}} & \multicolumn{1}{c}{\boldsymbol{x_E}} & \multicolumn{1}{c}{z} & \multicolumn{1}{c}{} \\ \cline{2-5} & \boldsymbol{A} & \boldsymbol{I} & \mathbf{0} & \boldsymbol{b} \\ \cline{2-5} & -\boldsymbol{c_D} & \mathbf{0} & 1 & 0 \\ \cline{2-5} \end{array}$$

Tableau du simplexe

- **Pour une base quelconque, on a :**

$$\hat{B} = \left(\begin{array}{c|c} B & 0 \\ \hline -c_B & 1 \end{array}\right) \Rightarrow \hat{B}^{-1} = \left(\begin{array}{c|c} B^{-1} & 0 \\ \hline c_B B^{-1} & 1 \end{array}\right)$$

$$T_0 = \begin{array}{c} \\ \\ \end{array}\begin{array}{|c|c|c||c|} \multicolumn{1}{c}{x_D} & \multicolumn{1}{c}{x_E} & \multicolumn{1}{c}{z} & \multicolumn{1}{c}{} \\ \hline A & I & 0 & b \\ \hline -c_D & 0 & 1 & 0 \\ \hline \end{array}$$

- **D'où le tableau:**

$$T_{\hat{B}} = \hat{B}^{-1} T_0 = \begin{array}{|c|c|c||c|} \multicolumn{1}{c}{x_D} & \multicolumn{1}{c}{x_E} & \multicolumn{1}{c}{z} & \multicolumn{1}{c}{} \\ \hline B^{-1}A & B^{-1} & 0 & B^{-1}b \\ \hline c_B B^{-1}A - c_D & c_B B^{-1} & 1 & c_B B^{-1} b \\ \hline \end{array}$$

$$T_{\hat{B}} \overset{P}{=} \begin{array}{|c|c|c||c|} \multicolumn{1}{c}{x_B} & \multicolumn{1}{c}{x_N} & \multicolumn{1}{c}{z} & \multicolumn{1}{c}{} \\ \hline I & -d_B \quad B^{-1}N & 0 & B^{-1}b \\ \hline 0 & c_B B^{-1}N - c_N & 1 & c_B B^{-1} b \\ \hline \end{array}$$

Valeurs des Variables de base

Coûts réduits

Valeur de z pour la solution de base

- **Tableau admissible si $B^{-1}b \geq 0$**

Le pivotage

- **Le pivotage permet de passer d'un tableau exprimé dans une base B1 à un autre tableau exprimé dans une base B2 adjacente**
- **Le pivotage définit une relation de voisinage entre les bases**

  Deux bases sont dites voisines base (ou adjacente) si elle ne diffère que par une et une seule variable de base.
- **Terminologie**
  - Si r est l'indice de la variable qui entre en base, la colonne $B^{-1}A_r$ est appelée colonne du pivot.
  - Si la variable de base B(i) sort de la base, la ligne i du tableau est appelée ligne du pivot.
  - L'élément α_{ir} qui se trouve sur la ligne du pivot et la colonne du pivot est appelé le pivot
- **Pivoter autour du pivot α_{ir} revient à**
  - Isoler la variable x_r dans la i^e équation en divisant la i^e ligne par $1/\alpha_{ir}$
  - Éliminer x_r des autres équations par substitution, cad en soustrayant aux autres lignes des multiples de la i^e ligne

Signatures

- **Signature d'un tableau Optimal**
  - Les coûts réduits sont tous positifs ou nuls et la solution de base est admissible

$$
\begin{array}{cccc|c||c}
x_1 & \dots & \dots & x_{n+m} & z & \\
\hline
 & & & & 0 & \oplus \\
 & & & & \vdots & \vdots \\
 & & & & 0 & \oplus \\
\hline
\oplus & \dots & \dots & \oplus & 1 & * \\
\hline
\end{array}
$$

- **Signature d'un tableau non-borné**
  - Il existe une variable hors base pour laquelle toutes les composantes de la distance sont positives

$$
\begin{array}{ccccc|c||c}
x_1 & \dots & x_k & \dots & x_{n+m} & z & \\
\hline
 & & \ominus & & & 0 & \oplus \\
 & & \ominus & & & \vdots & \vdots \\
 & & \ominus & & & 0 & \oplus \\
\hline
* & \dots & - & \dots & * & 1 & * \\
\hline
\end{array}
$$

Algorithme du simplexe (phase II)

- **Paramètres de l'algo**
  - Entrée = **un tableau admissible**
  - Sortie = **un tableau optimal ou non borné**

$$T_B \stackrel{P}{=} \begin{array}{c|c|c||c}
x_B & x_N & z & \\
\hline
I & B^{-1}N & 0 & \beta \\
\hline
0 & -\gamma_N & 1 & \zeta
\end{array}$$

- **Étapes**
  1) Choix d'une colonne (variable) entrante :
     Choisir une colonne hors base *r* ayant un coût réduit γ_r négatif
     S'il n'existe pas de colonne entrante : STOP le tableau courant est optimal
  2) Choix d'une colonne (variable) sortante :
     Choisir une ligne *j* minimisant le quotient caractéristique
     $\beta_j / \alpha_{jr} = \min_{k \in 1..m}(\beta_k / \alpha_{kr}) = \theta^*$
     S'il n'existe pas de colonne sortante : STOP le tableau courant est non borné
  3) Mise à jour de la base et du tableau
     Pivoter autour de α_{jr} et retourner en 1)

REM : en cas de conflit dans les choix de *j* et *r* appliquer la règle « du plus petit indice (règle de Bland)

Exemple 1

- Problème
  - **Une entreprise produit des câbles de cuivre de 5 et 10 mm de diamètre sur une seule ligne de production imposant les contraintes suivantes. On sait que :**
    - ***Le cuivre disponible permet de produire 21 000 mètres de câble de 5 mm de diamètre par semaine.***
    - ***Un mètre de câble de 10 mm de diamètre nécessite 4 fois plus de cuivre qu'un mètre de câble de 5 mm de diamètre.***
    - ***La demande hebdomadaire de câble de 5 mm est limitée a 15 000 mètres***
    - ***La production de câble de 10 mm ne dépasse pas 40% de la production totale***
    - ***Les câbles sont vendus respectivement 50 € et 200 € le mètre.***
  - **Maximiser le profit !**
- Modélisation
  - 2 variables de décision
    - *x1 : nb de milliers de mètres de câble de 5 mm produits chaque semaine*
    - *x2 : nb de milliers de mètres de câble de 10 mm produits chaque semaine*

Exemple 1

- **Forme canonique**

$$
\begin{array}{lrcrcl}
\text{Max} & z = 50000x_1 & + & 200000x_2 & & \\
\text{s.c.} & x_1 & + & 4x_2 & \leq & 21 \\
 & -4x_1 & + & 6x_2 & \leq & 0 \\
 & x_1 & & & \leq & 15 \\
 & x_1 & , & x_2 & \geq & 0
\end{array}
$$

- **Forme standard**

$$
\text{Max} \quad (z, \text{ s.c. } x_1, x_2, x_3, x_4, x_5 \geq 0)
$$

$$
\text{avec} \quad
\begin{array}{rcrcrcr}
x_3 & = & 21 & - & x_1 & - & 4x_2 \\
x_4 & = & 0 & + & 4x_1 & - & 6x_2 \\
x_5 & = & 15 & - & x_1 & & \\
\hline
z & = & 0 & + & 50000x_1 & + & 200000x_2
\end{array}
$$

Exemple 1

Première itération du simplexe

$$T_0 = \begin{array}{c} \\ \\ \\ \\ \\ \end{array}\begin{array}{cccccc|c|c} x_1 & x_2 & x_3 & x_4 & x_5 & z & & \text{q.c.} \\ \hline 1 & 4 & 1 & 0 & 0 & 0 & 21 & 21 \\ -4 & 6 & 0 & 1 & 0 & 0 & 0 & - \\ \color{orange}{1} & 0 & 0 & 0 & 1 & 0 & 15 & 15 \;\leftarrow \\ \hline -50000 & -200000 & 0 & 0 & 0 & 1 & 0 & \\ \uparrow & & & & & & & \end{array}$$

Deuxième itération du simplexe

$$T_1 = \begin{array}{cccccc|c|c} x_1 & x_2 & x_3 & x_4 & x_5 & z & & \text{q.c.} \\ \hline 0 & \color{orange}{4} & 1 & 0 & -1 & 0 & 6 & 3/2 \;\leftarrow \\ 0 & 6 & 0 & 1 & 4 & 0 & 60 & 10 \\ 1 & 0 & 0 & 0 & 1 & 0 & 15 & \infty \\ \hline 0 & -200000 & 0 & 0 & 50000 & 1 & 750000 & \\ & \uparrow & & & & & & \end{array}$$

Exemple 1

$$T_2 = \begin{array}{cccccc|c} x_1 & x_2 & x_3 & x_4 & x_5 & z & \\ \hline 0 & 1 & 1/4 & 0 & -1/4 & 0 & 3/2 \\ 0 & 0 & -3/2 & 1 & 11/2 & 0 & 51 \\ 1 & 0 & 0 & 0 & 1 & 0 & 15 \\ \hline 0 & 0 & 50000 & 0 & 0 & 1 & 1050000 \end{array}$$

- **Le tableau est optimal et la solution est :**

$$\begin{array}{ll} x_1^* = 15, & x_4^* = 51, \\ x_2^* = 1.5, & x_5^* = 0 \\ x_3^* = 0, & z^* = 1\,050\,000 \end{array}$$

Il faut donc fabriquer 15000 mètres de câble de cuivre de 5 mm de diamètre et 1500 m de câble de cuivre de 10mm pour un c.a. de 1 050 000 €

Exemple 2 (règle de Bland)

- **Forme canonique**

$$
\begin{array}{llrcrcr}
\text{Max} & z = & 2x_1 & + & 3x_2 & & \\
\text{s.c.} & & 2x_1 & - & x_2 & \leq & 6 \\
 & & x_1 & - & x_2 & \leq & 2 \\
 & & x_1 & & & \leq & 4 \\
 & & 4x_1 & - & x_2 & \leq & 14 \\
 & & & & x_2 & \leq & 5 \\
 & & x_1 & , & x_2 & \geq & 0
\end{array}
$$

- **Forme standard**

$$
\text{Max} \quad (z, \ \text{s.c.}\ x_1, x_2, x_3, x_4, x_5, x_6, x_7 \geq 0)
$$

$$
\text{avec} \quad
\begin{array}{rcrcrcr}
x_3 & = & 6 & - & 2x_1 & + & x_2 \\
x_4 & = & 2 & - & x_1 & + & x_2 \\
x_5 & = & 4 & - & x_1 & & \\
x_6 & = & 14 & - & 4x_1 & + & x_2 \\
x_7 & = & 5 & & & - & x_2 \\
\hline
z & = & 0 & + & 2x_1 & + & 3x_2
\end{array}
$$

Exemple 2 (règle de Bland)

$$T_0 = \begin{array}{ccccccc|c||c} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & z & \\ \hline 2 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 6 \\ 1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4 \\ 4 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 14 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 5 \\ \hline -2 & -3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ \uparrow & & & & & & & & \end{array} \quad \begin{array}{cc} \text{q.c.} & \\ 3 & \\ 2 & \leftarrow \\ 4 & \\ 7/2 & \\ \infty & \\ & \\ & \end{array}$$

$$T_1 = \begin{array}{ccccccc|c||c} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & z & \\ \hline 0 & 1 & 1 & -2 & 0 & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & -1 & 1 & 0 & 0 & 0 & 2 \\ 0 & 3 & 0 & -4 & 0 & 1 & 0 & 0 & 6 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 5 \\ \hline 0 & -5 & 0 & 2 & 0 & 0 & 0 & 1 & 4 \\ & \uparrow & & & & & & & \end{array} \quad \begin{array}{cc} \text{q.c.} & \\ 2 & \leftarrow \\ - & \\ 2 & \\ 2 & \\ 5 & \\ & \\ & \end{array}$$

Exemple 2 (règle de Bland)

$T_2 =$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	z		q.c.	
	0	1	1	-2	0	0	0	0	2	-	
	1	0	1	-1	0	0	0	0	4	-	
	0	0	-1	**1**	1	0	0	0	0	0	←
	0	0	-3	2	0	1	0	0	0	0	
	0	0	-1	2	0	0	1	0	3	3/2	
	0	0	5	-8	0	0	0	1	14		
				↑							

$T_3 =$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	z		q.c.	
	0	1	-1	0	2	0	0	0	2	-	
	1	0	0	0	1	0	0	0	4	∞	
	0	0	-1	1	1	0	0	0	0	-	
	0	0	-1	0	-2	1	0	0	0	-	
	0	0	**1**	0	-2	0	1	0	3	3	←
	0	0	-3	0	8	0	0	1	14		
			↑								

Exemple 2 (règle de Bland)

$$T_4 = \begin{array}{cccccccc|c} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & z & \\ \hline 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 5 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 1 & -1 & 0 & 1 & 0 & 3 \\ 0 & 0 & 0 & 0 & -4 & 1 & 1 & 0 & 3 \\ 0 & 0 & 1 & 0 & -2 & 0 & 1 & 0 & 3 \\ \hline 0 & 0 & 0 & 0 & 2 & 0 & 3 & 1 & 23 \end{array}$$

- **Le tableau est optimal et la solution est :**

$$\begin{array}{ll} x_1^* = 4, & x_5^* = 0, \\ x_2^* = 5, & x_6^* = 3, \\ x_3^* = 3, & x_7^* = 0, \\ x_4^* = 3, & z^* = 23. \end{array}$$

V. Programmation Linéaire

V.4. Choix d'une base initiale admissible : phase I du simplexe

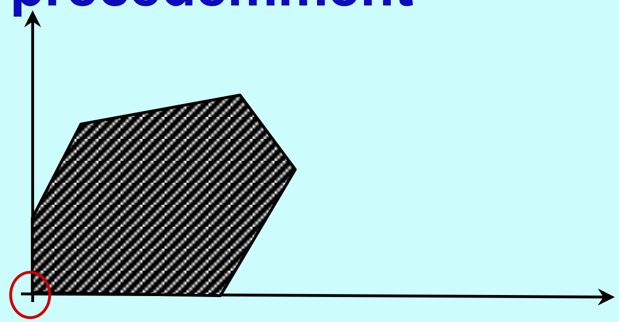
UNIVERSITÉ TOULOUSE III PAUL SABATIER

Cyril Briand (briand@laas.fr) – Master 2 AURO

Pourquoi une phase I ?

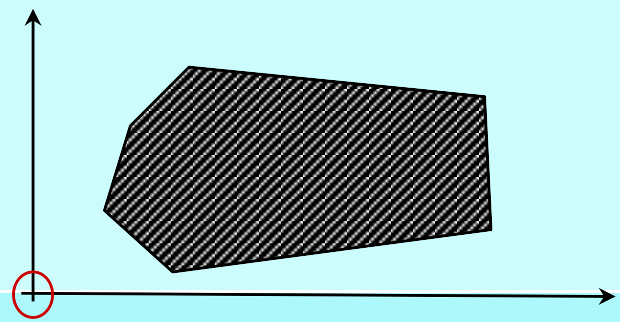
- **Algorithme du simplexe requiert x_0 = une solution de base initiale admissible**
  - Comment déterminer x_0 ?
  - Comment déterminer un tableau initial admissible?
- **C'est le rôle de la Phase I**

- **Cas simple : cas des exemples traités précédemment**
  - La solution de base initiale (x_D=0 et x_E=b) est admissible ($x_E \geq 0$)



- **Cas difficile**
  - Dans la solution de base initiale (x_D=0 et x_E=b) certaines var. d'écarts sont < 0 => non adm

L'origine n'appartient pas
au domaine admissible



Principes

- **On considère la forme standard d'un problème P1**

$$P1 = \begin{cases} \text{Max} & z = \boldsymbol{cx} \\ \text{s.c.} & \boldsymbol{Ax} = \boldsymbol{b} \\ & \boldsymbol{x} \geq 0 \end{cases}$$

- **Idée :**
  - Si la solution de base initiale (x_D=0 et X_E=b) n'est pas admissible (certaines composantes de b sont <0), alors résoudre un problème auxiliaire **P2** tel que :
    - *il soit lié au problème initial et*
    - *il soit trivial d'identifier une solution de base admissible pour ce problème auxiliaire.*
  - Pour cela, on introduit une variable auxiliaire x_0, et on modifie la fonction de coût.

$$P2 = \begin{cases} \text{Min} \quad z = & x_0 \\ & \sum_{b_i \geq 0, j=1}^{n+m} a_{ij}.x = b_i \\ & \sum_{b_i < 0, j=1}^{n+m} a_{ij}.x - x_0 = b_i \\ & x, x_0 \geq 0 \end{cases}$$

Propriétés

$$
P2 = \begin{cases}
\text{Max} \quad z = -x_0 \\
\sum_{b_i \geq 0,\, j=1}^{n+m} a_{ij}.x = b_i \\
\sum_{b_i < 0,\, j=1}^{n+m} a_{ij}.x - x_0 = b_i \\
x, x_0 \geq 0
\end{cases}
$$

- **Le coût optimal de P2 ne peut être > 0 ($x_0 \geq 0$)**
- **Une solution de base admissible de P2 peut être facilement trouvée en faisant rentrer la variable x_0 en base et en faisant sortir la variable d'écart la plus négative**
  - toutes les autres variables d'écarts restant en base deviendront forcément positive
  - On peut alors appliquer l'algorithme du simplexe
- **Supposons que x* soit une solution admissible de P1.**
  - Dans ce cas, $Ax^* \leq b$ et $x^* \geq 0$.
  - $x=x^*$ et $x_0=0$ est une solution admissible de P2.
  - Le coût associé est 0. C'est donc une solution optimale de P2.
- **D'où**
  - Si P1 possède une solution admissible alors le coût optimal de P2 est 0.
- **Contraposée :**
  - Si le coût optimal de P2 est strictement négatif alors P1 ne possède pas de solution admissible.

Algorithme du simplexe à deux phases

- **Phase I**

1. Introduire la variable artificielles x_0, et la faire renter en base en faisant sortir la variable d'écart la plus négative
2. Appliquer la méthode du simplexe au problème auxiliaire
3. Si le coût optimal est strictement négatif, le problème original n'est pas admissible. STOP.

- **Phase II**

1. Supprimer les colonnes du tableau correspondant aux variables artificielles
2. En fonction de la solution de base admissible génére lors de la phase I, recalculer la ligne des coûts
3. **Appliquer la méthode du simplexe au tableau obtenu.**

- **Remarques**
  - L'algo est complet car il peut gérer toutes les issues possibles
  - Si le cyclage est empêché (par la règle de Bland), il y a 3 possibilités

Le problème n'est pas admissible. Détecté à la fin de la phase I.

Le coût optimal est $+\infty$. Détecté lors de la phase II.

La phase II se termine avec une solution optimale.

Exemple

MAX Z = x1 + 2x2
subject to
-x1 - x2 <= -20
x1 - x2 <= 40
-x1 + x2 <= 40
x1 + x2 <= 80
-x1 - 2x2 <= -30
and x1,x2 >= 0

VI. Programmation Linéaire en Nombres Entiers (PLNE)

VI.1. Introduction

UNIVERSITÉ TOULOUSE III PAUL SABATIER

Cyril Briand (briand@laas.fr) – Master 2 AURO

Définitions

- Un programme en nombres entiers est un programme dont les variables sont contraintes à ne prendre que des valeurs entières.

- Souvent, elles sont contraintes à prendre les valeurs 0 ou 1. On parle alors d'un programme binaire.

- Un programme mixte en nombres entiers est un programme dont <u>certaines</u> variables sont contraintes à ne prendre que des valeurs entières.

$$\min c^T x + d^T y + e^T z$$

s.c.

$$Ax + By + Cz = b$$
$$x, y, z \geq 0$$
$$y \text{ entier}$$
$$z \in \{0,1\}$$

Intérêts et inconvénients de la PLNE

- **Augmentation du champ d'application de la PL**
  - Gestion des cas où les variables doivent être nécessairement entières (nb personnes allouées à un travail, nb camions utilisés pour une collecte de déchets…)
  - De nombreuses contraintes logiques, en apparence non linéaires, peuvent être modélisées, à l'aide de variables entières
    - *Exemple : problème d'optimisation où on doit choisir entre 2 actions* A *et* B *exclusives. L'exclusion est modélisée par la contrainte* $x_A + x_B <= 1$ *où* x_A , x_B *sont des variables binaires*

- **Méthodes de résolution génériques et efficaces pour des problèmes d'optimisation combinatoire de taille raisonnable**
  - Des milliers de variables binaires / des centaines de variables entières
  - Rem : Il existe des programmes beaucoup plus gros résolus efficacement (la difficulté d'un programme ne dépend pas que du nb de variables et de contraintes)
  - Solveurs : Cplex, Gurobi, SCIP, …
- **En dehors de toute résolution, PLNE = puissant langage de modélisation d'un problème**
  - Le modèle permet de comprendre les propriétés du problème
  - Le modèle permet de communiquer sans ambiguïté

- **Cependant**
  - Pas de méthodes de résolution en temps polynomial (contrairement à la PL)
  - Pour beaucoup de problèmes, des techniques d'optimisation spécifiques sont plus efficaces

Relaxation linéaire

- **Soit un programme linéaire mixte en nombres entiers**

$$\min c^T x + d^T y + e^T z$$

s.c.

$$Ax + By + Cz = b$$
$$x, y, z \geq 0$$
$$y \text{ entier}$$
$$z \in \{0,1\}$$

- **Le programme linéaire où on relâche la contrainte d'intégrité**

$$\min c^T x + d^T y + e^T z$$

s.c.

$$Ax + By + Cz = b$$
$$x, y \geq 0$$
$$0 \leq z \leq 1$$

est appelé sa relaxation linéaire.

- **Remarques**
  - Si la solution optimale du problème relâché est entière, elle est optimale pour le problème de départ
  - Sinon elle indique une borne inférieure du critère (supérieure dans le cas d'un problème de maximisation)
  - Le polyèdre admissible associé au problème relâché peut être non vide alors que le problème initial n'a aucune solution

Difficulté de la PLNE

- **Pourquoi ne pas tout simplement arrondir, à l'entier le plus proche, la solution optimale de la relaxation linéaire ???**

- **Ça peut marcher …**
  - Quand les variables représentent des quantités importantes
    - *Exemple*
      - Production de plusieurs milliers de stylos-billes
      - Si une contrainte est légèrement violée du fait de l'arrondi, ça ne porte pas à conséquence
    - *Mais ce n'est pas toujours évident d'arrondir…*
  - Quand le PL relâché est équivalent au PL initial
    - *Exemple : problème de transport*
    - *C'est rare …*
    - *Mais quand c'est le cas, on est heureux !*

- **Le plus souvent ça ne marche pas !**
  - Illustration 1
  - Illustration 2

Illustration 1

- **Objectifs : Montrer que ..**
  - L'entier le plus proche n'est pas forcément réalisable …
  - Le plus proche entier réalisable n'est pas forcément optimal

- **Exemple**

$$\max 3x_1 + 13x_2$$

sc.

$$2x_1 + 9x_2 \leq 40$$

$$11x_1 - 8x_2 \leq 82$$

$$x_1, x_2 \geq 0$$

$$x_1, x_2 \text{ entiers}$$

Illustration 1

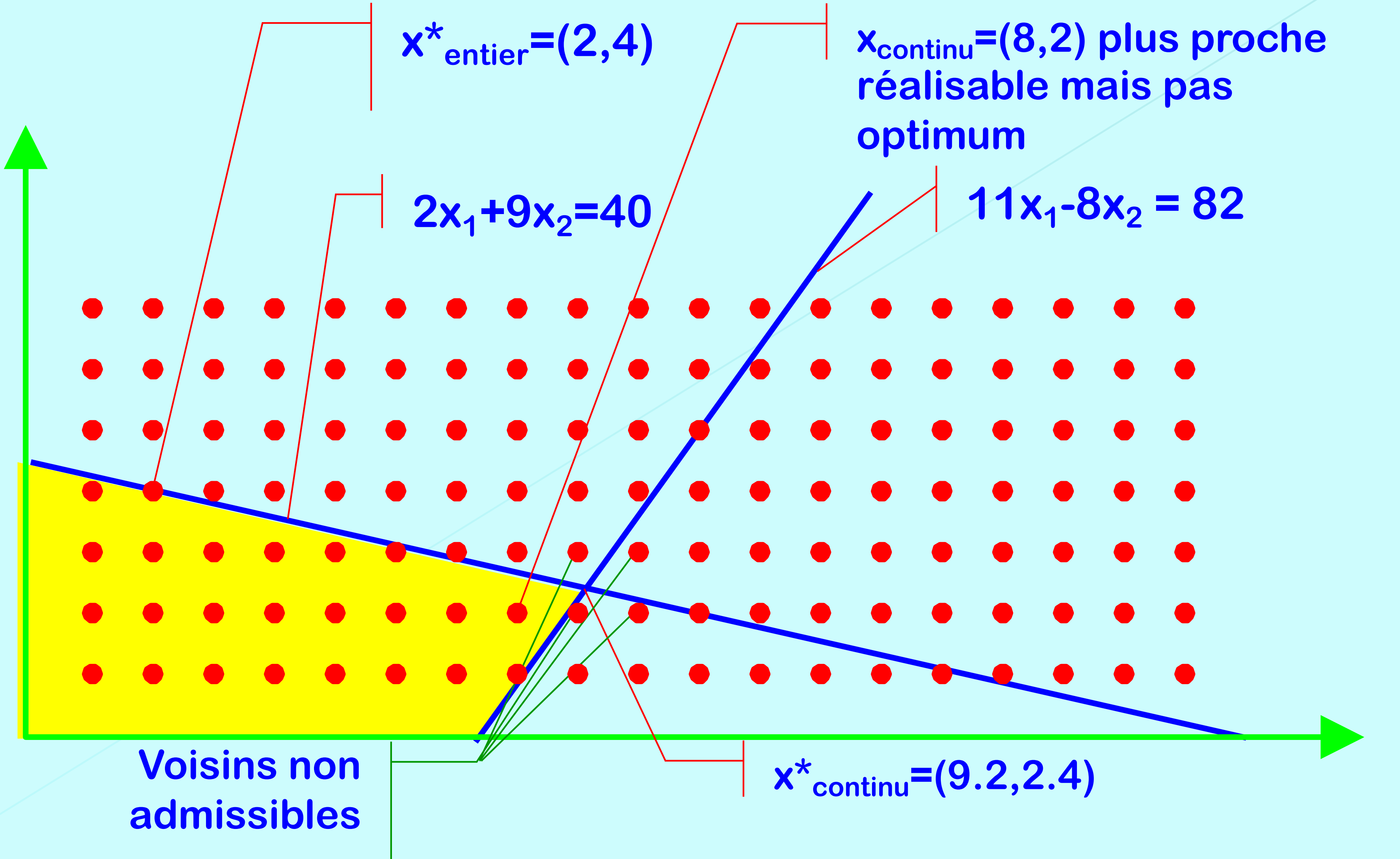


Illustration 2

- **Objectif : Montrer que …**
  - Arrondir une variable de décision qui vaut 0.5 à 0 ou à 1 n'a pas de sens..

- **Exemple**
  - Une société dispose de 1 400 000 € à investir.
  - Les experts proposent 4 investissements possibles

	Coût	Bénéfice	Rendement
Inv. 1	500 000	1 600 000	3.20
Inv. 2	700 000	2 200 000	3.14
Inv. 3	400 000	1 200 000	3.00
Inv. 4	300 000	800 000	2.67

Illustration 2

Modélisation :

- Variables de décision : x_i, i=1,…,4
- $x_i = 1$ si investissement i est choisi
- $x_i = 0$ sinon
- Objectif : maximiser bénéfice

$$\max 16\, x_1 + 22\, x_2 + 12\, x_3 + 8\, x_4$$

- Contrainte : budget d'investissement

$$5\, x_1 + 7\, x_2 + 4\, x_3 + 3\, x_4 \leq 14$$

Illustration 2

- **Première relaxation**

Programme linéaire 1

$$\max 16 x_1 + 22 x_2 + 12 x_3 + 8 x_4$$

s.c.

$$5 x_1 + 7 x_2 + 4 x_3 + 3 x_4 \leq 14$$

$$x_1, x_2, x_3, x_4 \geq 0$$

- **Solution :**

$$x_1 = 2.8,\ x_2 = 0,\ x_3 = 0,\ x_4 = 0$$

- **Interprétation ?**
  - Effectuer l'investissement 1.
  - Bénéfice : 1 600 000 €.
  - Coût : 500 000 €
- **Relaxation trop forte !**

Illustration 2

- **Deuxième relaxation**

Programme linéaire 1

$$\max 16\,x_1 + 22\,x_2 + 12\,x_3 + 8\,x_4$$

s.c.

$$5\,x_1 + 7\,x_2 + 4\,x_3 + 3\,x_4 \leq 14$$
$$x_1, x_2, x_3, x_4 \geq 0$$
$$x_1, x_2, x_3, x_4 \leq 1$$

- **Solution :**

$$x_1 = 1,\ x_2 = 1,\ x_3 = 0.5,\ x_4 = 0$$

- **Interprétation ?**
  - Effectuer les investissements 1 et 2.
  - Bénéfice : 3 800 000 €.
  - Coût : 1 200 000 €.
  - Plus assez de budget pour l'investissement 3.

Illustration 2

- **Pas de relaxation**

Programme en nombres entiers

$$\max 16\,x_1 + 22\,x_2 + 12\,x_3 + 8\,x_4$$

s.c.

$$5\,x_1 + 7\,x_2 + 4\,x_3 + 3\,x_4 \leq 14$$
$$x_1, x_2, x_3, x_4 \in \{0,1\}$$

- **Solution :**

$$x_1 = 0,\ x_2 = 1,\ x_3 = 1,\ x_4 = 1$$

- **Interprétation ?**
  - Solution évidente à interpréter : c'est la meilleure!
  - Bénéfice : 4 200 000 €.
  - Coût : 1 400 000 €
- **Conclusion**
  - Dans l'exemple, l'investissement 1 est le plus rentable, mais il n'est pas repris dans la solution optimale.
  - Les contraintes d'intégralité peuvent modifier significativement la structure de la solution.

Méthode de résolution des PLNE

- **Résolution comme un PL ordinaire**
  - Cas où PLNE équivalent à leur problème relaxé
- **Méthodes arborescentes**
  - Aussi appelées méthodes de séparation et d'évaluation (branch-and-bound)
  - Principe :
    - *Séparer chaque problème (nœud de l'arborescence) en deux sous-problèmes où une variable x est posée soir inférieure ou égale à une valeur entière k, soit strictement supérieure à cette même valeur*
    - *Évaluer chaque nouveau nœud*
    - *Ne développer que les nœuds les plus prometteurs*
- **Méthodes de coupes (ou de troncature)**
  - Principe : tenter de réduire le domaine du PL relaxé en ajoutant de nouvelles contraintes de sorte à déterminer l'enveloppe convexe du PLNE
  - *Non-étudiées ici*
- **Méthodes de génération de colonnes**
  - Méthodes utilisées pour des problèmes de grande taille
  - *Non-étudiées ici*

VI. Programmation Linéaire en Nombres Entiers (PLNE)

VI.2. Cas des PLNE équivalents à leur PL relaxé

UNIVERSITÉ TOULOUSE III PAUL SABATIER

Cyril Briand (briand@laas.fr) – Master 2 AURO

Le théorème des valeurs entières

- **Théorème**
  - **Soit le problème**

$$\begin{array}{llrcl} \text{Opt} & z = & cx & & \\ \text{s.c.} & & Ax & = & b \\ & & x & \geq & 0 \end{array}$$

où A est une matrice m x n de rang m a coefficients entiers et b est un vecteur entier.

Les affirmations suivantes sont équivalentes :

- ***le déterminant de toute base B de A vaut +1 ou -1;***
- ***pour tout b entier, le polyèdre $P = \{x \mid Ax = b;\ x \geq 0\}$ est entier;***
- ***pour toute base B de A, B^{-1} est entière.***

Matrice totalement unimodulaire

- **Définition**
  - **Une matrice A est totalement unimodulaire si toute sous-matrice carrée que l'on peut extraire de A (en éliminant des lignes et des colonnes) a un déterminant égal a 0, 1 ou -1**

- **Théorème**
  - **Soit un PL où *A* est totalement unimodulaire. Pour tout vecteur *b* entier, le problème admet une solution admissible entière dés qu'il a une solution admissible. De plus, il admet une solution optimale entière des qu'il a une solution optimale.**

- **Condition suffisante d'unimodularité – *Heller et Tomkins***
  - Une matrice A est totalement unimodulaire si elle satisfait les conditions suivantes
    - chaque terme vaut 0, 1 ou -1;
    - chaque colonne contient au plus deux éléments non nuls ;
    - l'ensemble I des lignes de A peut être partitionné en deux ensembles I1 et I2 tels que
      - deux termes non nuls de même signe d'une colonne sont l'un dans une ligne de I1, l'autre dans une ligne de I2,
      - deux termes non nuls de signe opposé d'une colonne sont tous les deux dans des lignes de I1 ou tous les deux dans des lignes de I2.

Problèmes de réseaux

- **Propriétés**
  - Soit un graphe G comprenant *m* nœuds et *n* arcs reliant chacun un nœud à un autre.
  - La matrice d'incidence sommets-arcs d'un graphe est totalement unimodulaire
  - Preuve (condition suffisante d'*Heller et Tomkins* vérifiée)
    - *La matrice d'incidence sommets-arcs de ce graphe a m lignes et n colonnes*
    - *Si l'arc en colonne k va du nœud i au nœud j alors A(i,k)=1 et A(j,k)=-1 et les autres coefficients sont nuls*
    - *Prendre I1=I et I2=ø*

- **La plupart des problèmes de réseaux peuvent donc être résolus par la PL**
  - Cela ouvre la porte à des techniques de résolutions génériques mais …
  - … elles ne sont pas toujours les plus performantes

Exemple : calcul du flot maximum

- **On peut formuler le problème comme un Programme Linéaire**
  - Matrice des contraintes = Matrice d'incidence sommets-arcs
    - *matrice totalement unimodulaire ➔ L'algo du simplexe trouve toujours une solution optimale <u>entière</u>*

$$maximiser\ F$$

$$\sum_{j \in \Gamma(i)} \Phi_{ij} - \sum_{j \in \Gamma^{-1}(i)} \Phi_{ji} \left| \begin{array}{l} = F\ si\ x_i = s \\ \\ = 0\ si\ x_i \neq s,t \\ = -F\ si\ x_i = t \end{array} \right.$$

$$0 \leq \Phi_{ij} \leq C_{ij},\ \forall (x_i, x_j) \in X$$

VI. Programmation Linéaire en Nombres Entiers (PLE)

VI.3. Méthode arborescente

UNIVERSITÉ TOULOUSE III PAUL SABATIER

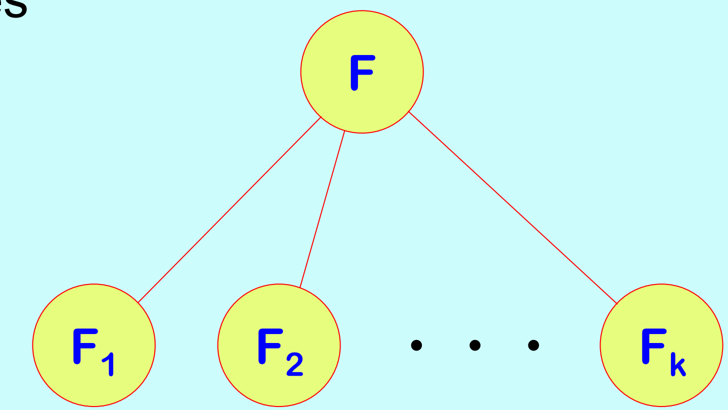
Cyril Briand (briand@laas.fr) – Master 2 AURO

Branch & Bound

- **Idées :**
  - Diviser pour conquérir
  - Utilisation de bornes sur le coût optimal pour éviter d'explorer certaines parties de l'ensemble des solutions admissibles.
- **Branch**
  - Soit F l'ensemble des solutions admissibles d'un problème :

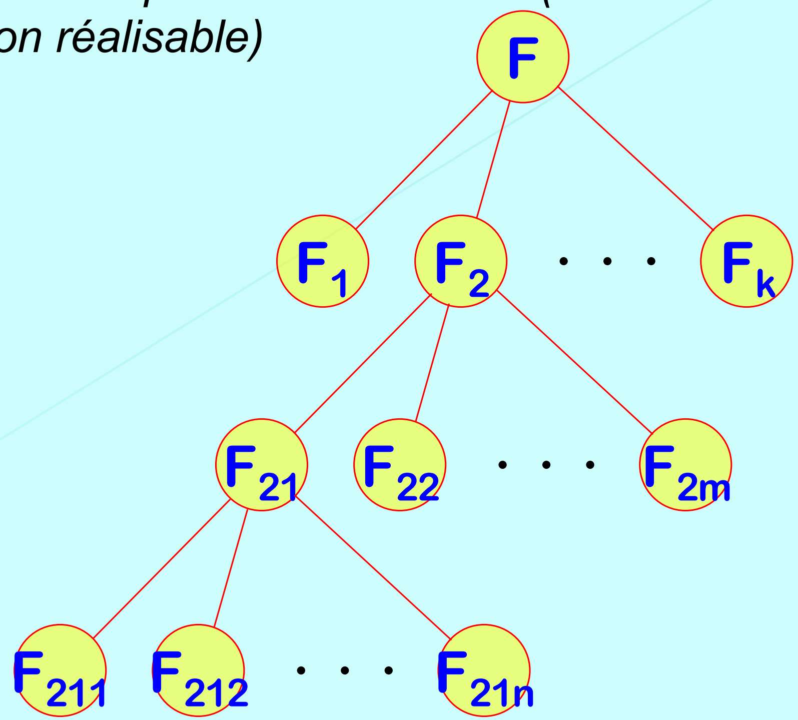
    $\min c^T x \quad$ s.c. $x \in F$
  - On partitionne F en un une collection finie de sous-ensembles $F_1,\ldots,F_k$.
  - On résout séparément les problèmes

    $\min c^T x \;$ s.c. $x \in F_i$
  - Par abus de langage, ce problème sera appelé F_i également.



Branch & Bound

- **Remarques**
  - A priori, les sous-problèmes peuvent être aussi difficiles que le problème original
  - Dans ce cas, on applique le même système
    - *On partitionne le/les sous-problèmes*
    - *Parcours en profondeur d'abord (← trouver rapidement une première solution réalisable)*



Branch & Bound

- **Bound**
  - On suppose que pour chaque sous-problème min c^Tx s.c. $x \in F_i$
  - on peut calculer efficacement une borne inférieure $b(F_i)$ sur le coût optimal, i.e.

$$b(F_i) \leq \min_{x \in F_i} c^T x$$

  - Typiquement, on utilise **la relaxation linéaire** pour obtenir cette borne (mais il existe d'autres types de relaxation)

- **Algorithme général**
  - A chaque instant, on maintient
    - *une liste de sous-problèmes actifs,*
    - *le coût U de la meilleure solution obtenue jusqu'alors.*
    - *Valeur initiale de U : soit ∞, soit c^Tx pour un x admissible connu.*

  - **Une étape typique** :
    1. *Sélectionner un sous-problème actif F_i*
    2. *Si F_i est non admissible, le supprimer. Sinon, calculer $b(F_i)$.*
    3. *Si $b(F_i) \geq U$, supprimer F_i.*
    4. *Si $b(Fi) < U$, soit résoudre F_i directement, soit créer de nouveaux sous-problèmes et les ajouter à la liste des sous-problèmes actifs.*

Principes de séparation

- **Comment séparer ?**
  - Si la solution de la relaxation est entière
    - *pas besoin de partitionner le sous-problème (sol optimale pour le nœud courant)*
  - Sinon
    - *on choisit un x^*_i non entier, a priori n'importe lequel*
    - *on crée deux sous-problèmes en ajoutant les contraintes :*
      $x_i \leq \lfloor x^*_i \rfloor$ *et* $x_i \geq \lceil x^*_i \rceil$ *(ces contraintes sont violées par x^*).*

- **Exemple**
  - Soit le problème F en forme canonique

$$\begin{aligned} & \min x_1 - 2x_2 \\ \text{s.c.} \quad & -4x_1 + 6x_2 \leq 9 \\ & x_1 + x_2 \leq 4 \\ & x_1, x_2 \geq 0 \\ & x_1, x_2 \text{ entiers} \end{aligned}$$

Exemple

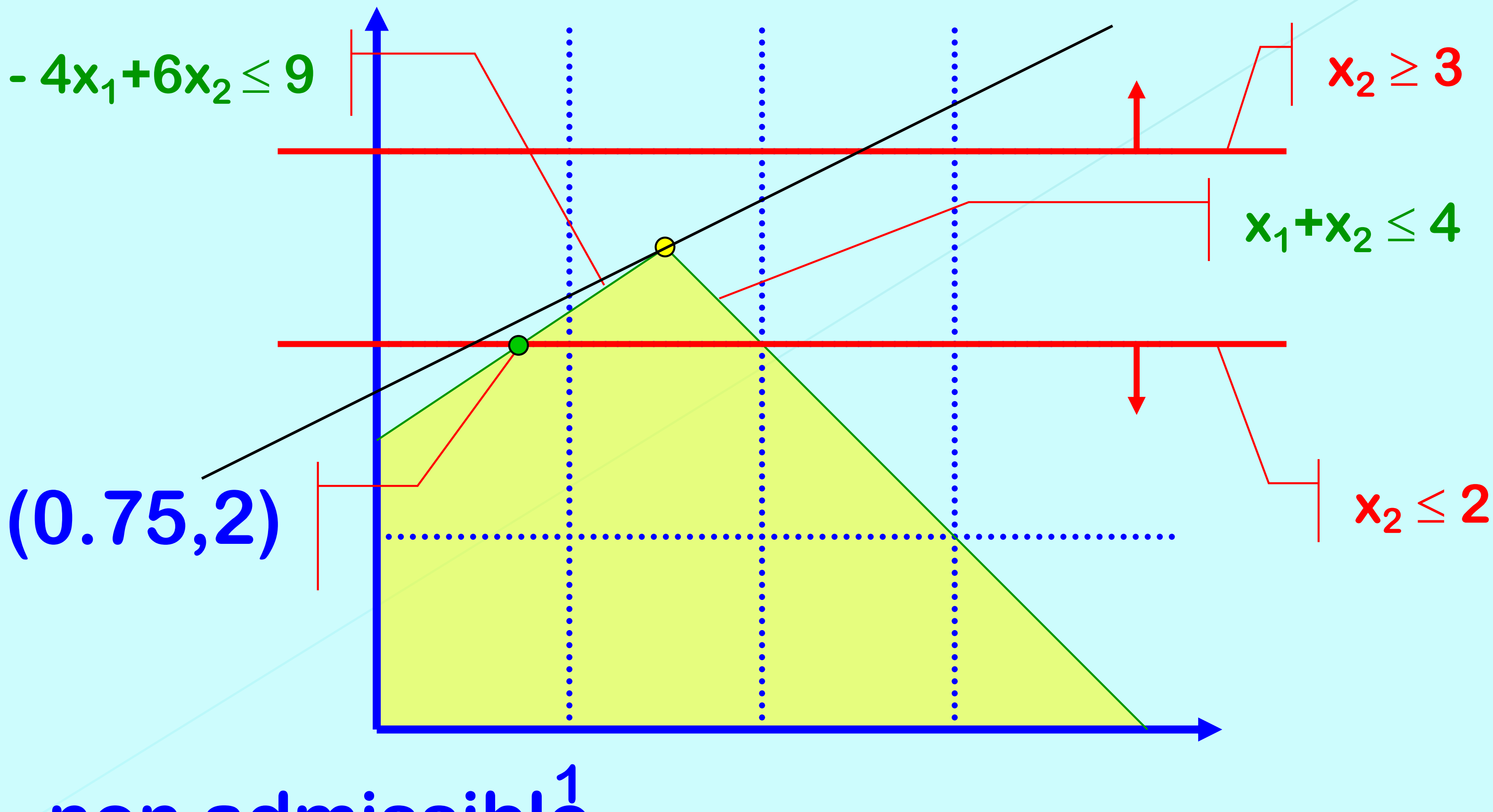
- **Première itération**
  - $U = +\infty$
  - Liste des sous-problèmes actifs : {F}
  - Solution de la relax. de F : $x^* = (1.5, 2.5)$
  - $b(F) = -3.5$
  - Création des sous-problèmes en rajoutant les contraintes

$$x_2 \leq \lfloor x^*_2 \rfloor = 2$$
$$x_2 \geq \lceil x^*_2 \rceil = 3$$

F_1	F_2
$\min\ x_1 - 2x_2$	$\min\ x_1 - 2x_2$
s.c. $-4x_1 + 6x_2 \leq 9$	s.c. $-4x_1 + 6x_2 \leq 9$
$x_1 + x_2 \leq 4$	$x_1 + x_2 \leq 4$
$x_1, x_2 \geq 0$	$x_1, x_2 \geq 0$
$x_2 \geq 3$	$x_2 \leq 2$
x_1, x_2 entiers	x_1, x_2 entiers

- Liste des sous-problèmes actifs : {F,F1,F2}

Exemple



- F_1 non admissible
- Liste des sous-problèmes actifs : $\{F,F_2\}$

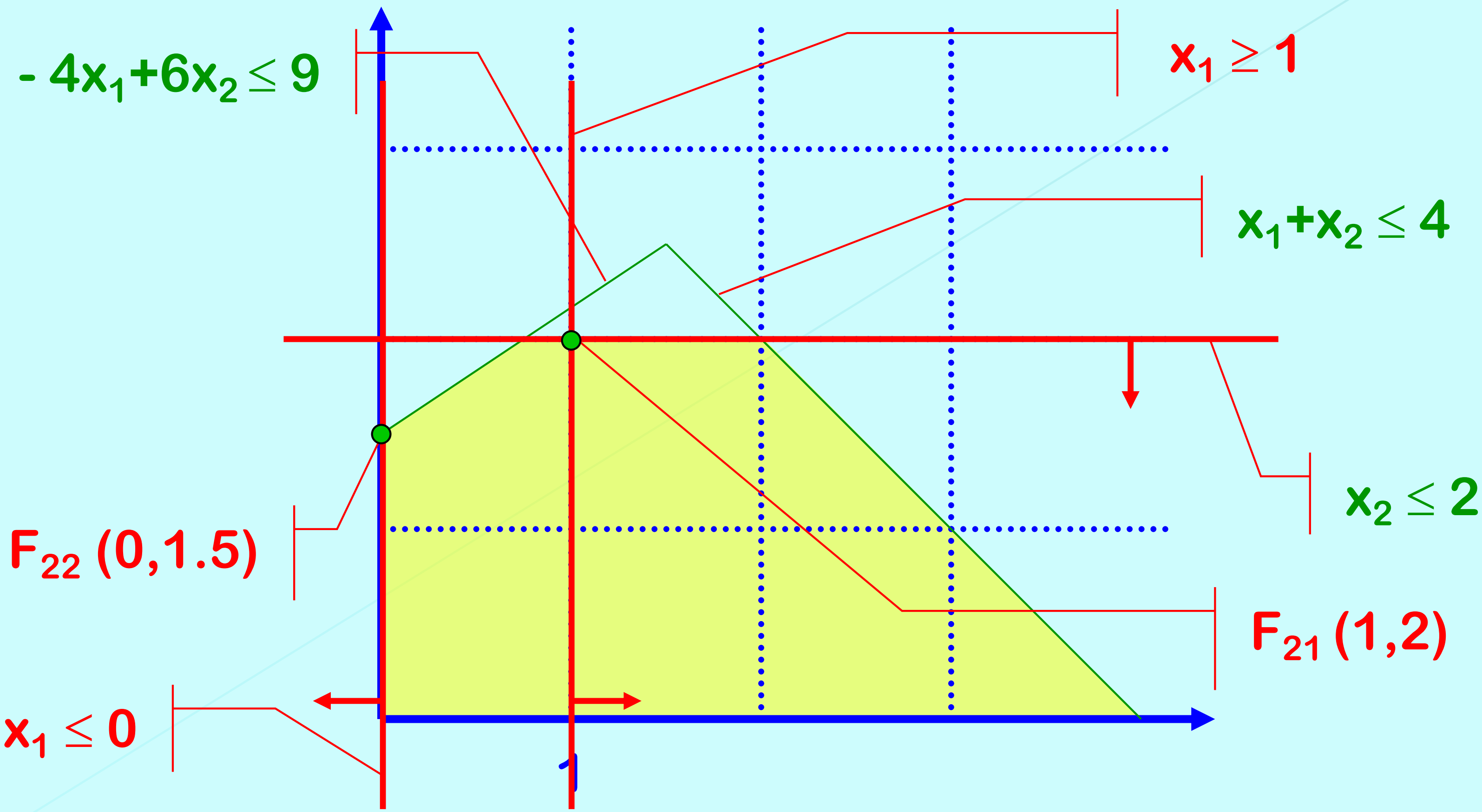
Exemple

- **Nouvelle itération**
  - $U = +\infty$
  - Liste des sous-problèmes actifs : $\{F, F_2\}$
  - Solution de la relax. de F_2 : $x^* = (0.75, 2)$
  - $b(F_2) = -3.25$
  - Création des sous-problèmes en rajoutant les contraintes

$$x_1 \leq \lfloor x^*_1 \rfloor = 0$$
$$x_1 \geq \lceil x^*_1 \rceil = 1$$

F_{21}	F_{22}
$\min x_1 - 2x_2$	$\min x_1 - 2x_2$
s.c. $-4x_1 + 6x_2 \leq 9$	s.c. $-4x_1 + 6x_2 \leq 9$
$x_1 + x_2 \leq 4$	$x_1 + x_2 \leq 4$
$x_1, x_2 \geq 0$	$x_1, x_2 \geq 0$
$x_2 \leq 2$	$x_2 \leq 2$
$x_1 \geq 1$	$x_1 \leq 0$
x_1, x_2 entiers	x_1, x_2 entiers

Exemple



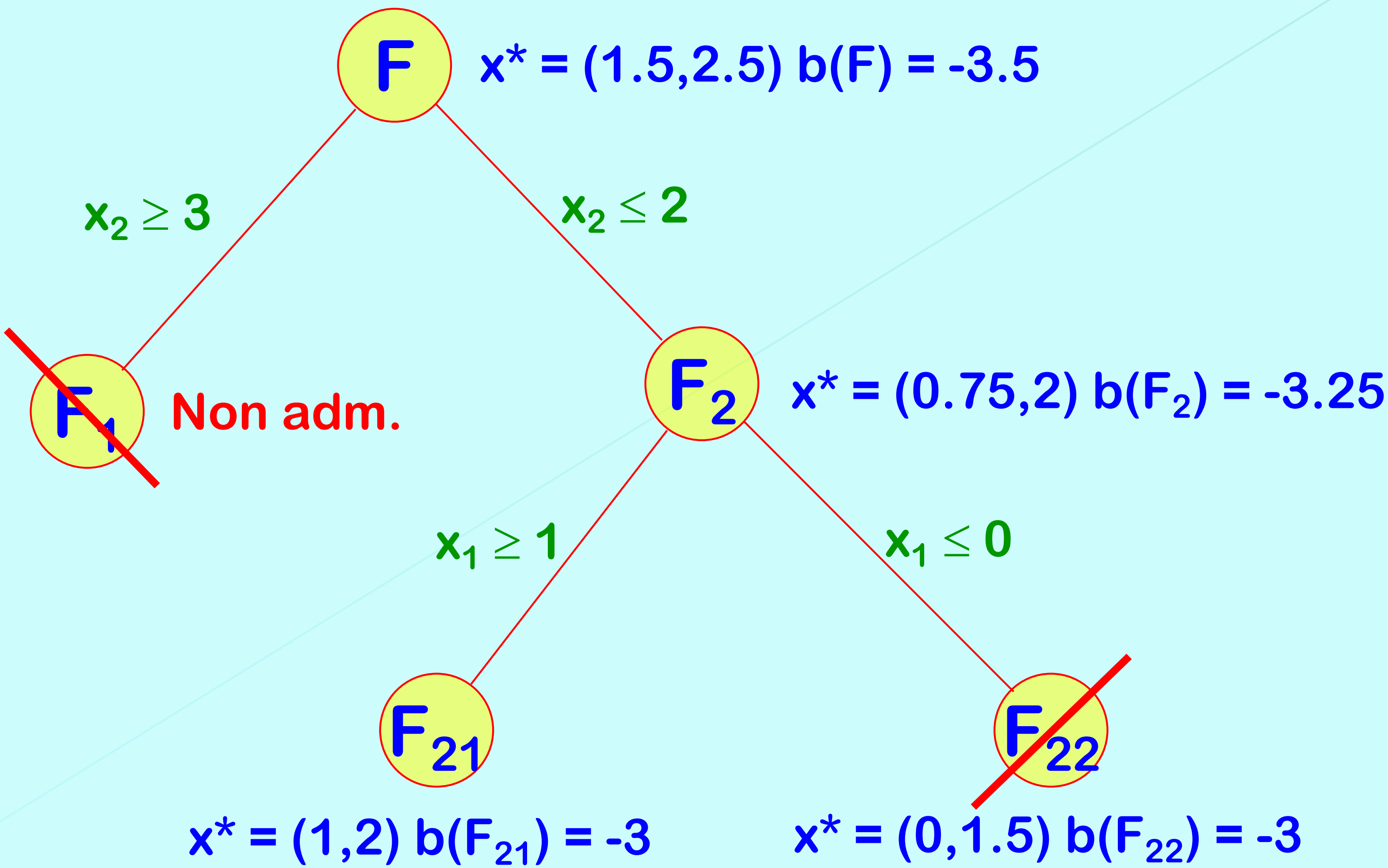
- Liste des sous-problèmes actifs : $\{F, F_2, F_{21}, F_{22}\}$

Exemple

- **Nouvelle itération**
  - $U = +\infty$
  - Liste des sous-problèmes actifs : $\{F, F_2, F_{21}, F_{22}\}$
  - Solution de la relax. de F_{21} : $x^* = (1,2)$
  - $(1,2)$ est solution de F_{21}
  - $b(F_{21}) = -3$
  - $U = -3$

- **Nouvelle itération**
  - $U = -3$
  - Liste des sous-problèmes actifs : $\{F, F_2, F_{22}\}$
  - Solution de la relax. de F_{22} : $x^* = (0,1.5)$
  - $b(F_{22}) = -3 \geq U$
  - Liste des sous-problèmes actifs : $\{F, F_2\}$
  - Solution de $F_2 = (1,2)$
  - Solution de $F = (1,2)$

Exemple


F
$x^* = (1.5,2.5)$ $b(F) = -3.5$
$x_2 \geq 3$
$x_2 \leq 2$
F_1
Non adm.
F_2
$x^* = (0.75,2)$ $b(F_2) = -3.25$
$x_1 \geq 1$
$x_1 \leq 0$
F_{21}
F_{22}
$x^* = (1,2)$ $b(F_{21}) = -3$
$x^* = (0,1.5)$ $b(F_{22}) = -3$

VI. Programmation Linéaire en Nombres Entiers (PLNE)

VI.4. Modélisation

UNIVERSITÉ TOULOUSE III PAUL SABATIER

Cyril Briand (briand@laas.fr) – Master 2 AURO

PL-01 : modélisation (k parmi n, implication)

- **Dans les problèmes pratiques, il existe de nombreuses contraintes logiques**
  - Ces contraintes peuvent être prises en compte plus ou moins astucieusement à l'aide de de contraintes ou de variables nouvelles

- **Illustrons ceci sur le problème d'investissement**

  La société se voit imposer des contraintes supplémentaires.

  a) Elle peut opérer au maximum deux investissements

  Nouvelle contrainte: $x_1 + x_2 + x_3 + x_4 \leq 2$

  b) Si elle décide d'opérer l'invest. 2, elle doit aussi opérer l'invest. 1.

  Nouvelle contrainte : $x_2 - x_1 \leq 0$

  - Si $x_2 = 1$, alors $x_1 \geq 1$.Comme $x_1 \in \{0,1\}$, alors $x_1 = 1$.
  - Si $x_2 = 0$, alors $x_1 \geq 0$. Pas de restriction sur x_1.

  c) Si elle décide d'opérer l'invest. 2, elle ne peut pas opérer l'invest. 4.

  Nouvelle contrainte : $x_2 + x_4 \leq 1$

  - Si x2 = 1, alors x4 $\leq$ 0.Comme x4 $\in$ {0,1}, alors x4 = 0.
  - Si x2 = 0, alors x4 $\leq$ 1. Pas de restriction sur x4.

PLNE - Modélisation (coûts fixes)

d) On suppose que l'investissement x_1 peut être réalisé plusieurs fois (PL-01 => PLNE). Le cahier des charge impose que si l'investissement x_1 est réalisé au moins une fois alors un coût fixe doit être payé, à déduire des bénéfices réalisés proportionnels à x_1

- *C(x1) = 0 si x1=0 et C(x1)=cx1 – K sinon (x1>0)*
- *Ce n'est plus linéaire !!!*

- *Solution : introduire une nouvelle variable y (y=0 si x1=0 et y=1 si x1>0) et reécrire le coût*

  Max Cx1 –kY avec la contrainte

  x1 $\leq$By où B est une borne supérieure de x1
- *Remarque*
  - On a bien x>0 implique y=1 de par la contrainte
  - Par contre la contrainte n'impose pas x=0 => y=0 …
  - … mais le sens d'optimisation si!

PLNE Modélisation (disjonction)

- Soit - Soit
  - on veut garantir qu'au moins une des contraintes $f(x_1,\ldots,x_n) \leq 0$ ou $g(x_1,\ldots,x_n) \leq 0$ soit vérifiée, mais pas nécessairement les deux.
  - Solution
    - *Choisir B_1 et B_2 suffisamment grand pour que*

      $$f(x_1,\ldots,x_n) \leq B_1 \text{ et } g(x_1,\ldots,x_n) \leq B_2 \ \forall x$$
    - *Introduire une nouvelle variable binaire $y \in \{0,1\}$*
    - *Poser les contraintes suivantes :*

      $$f(x_1,\ldots,x_n) \leq B_1(1-y)$$

      $$g(x_1,\ldots,x_n) \leq B_2 y$$
    - *Si $y = 0$, alors les contraintes sont $f(x_1,\ldots,x_n) \leq B_1$ (toujours vraie) et $g(x_1,\ldots,x_n) \leq 0$*
    - *Si $y = 1$, alors les contraintes sont $f(x_1,\ldots,x_n) \leq 0$ et $g(x_1,\ldots,x_n) \leq B_2$ (toujours vraie)*
    - *Du point de vue algorithmique, il vaut mieux que les valeurs de B_1 et B_2 soient de "bonnes" bornes sup (cad les plus petites possibles)*

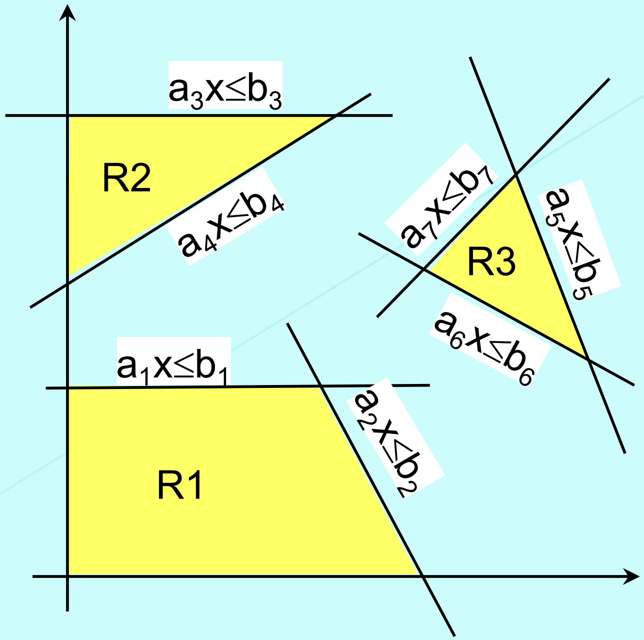
PLNE Modélisation (implication, k parmi n)

- **$f(x_1,\ldots,x_n) > 0 \Rightarrow g(x_1,\ldots,x_n) \leq 0$**
  - A => B s'écrit de façon logique $\neg A \vee B$

- **Au moins k contraintes parmi les n contraintes $f_1(x_1,\ldots,x_n)$ ... $f_n(x_1,\ldots,x_n) \leq 0$ doivent être vérifiées**

PLNE Modélisation (domaines non-convexes)

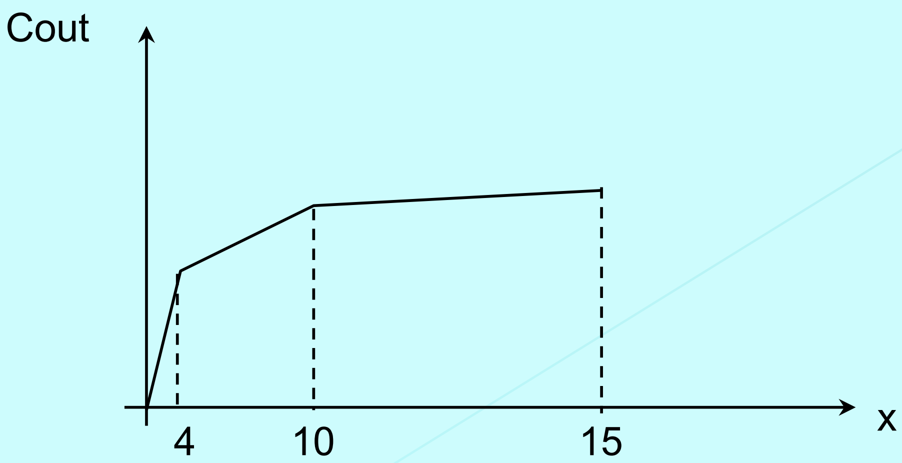
- **Un domaine réalisable qui consiste en l'union de régions, chacune définie par un système d'inégalités affines peut être modélisé sous forme PLNE**
  - Exemple



  - De même pour des régions qui ne sont pas disjointes (domaine polyèdral non-convexe)

Objectif = fonctions affines par morceaux

- **Exemple : coût d'expansion d'une usine avec 3 segments de pente 5, 3 et 1 par millier d'articles quotidien d'expansion**



- **De façon générale, on impose**

  $L_j y_j \leq x_j \leq L_j y_{j-1}$ où Lj est la longueur du jème segment

- **On peut approximer des fonctions non linéaires par des fonctions affines par morceaux …**
  - Plus le nb de segments est important, meilleure est l'approximation

Produits de variables

- Produits de variables binaires

- Produits de variables binaires avec une variable continue