Recherche opérationnelle et applications

Bernard Fortz

2012-2013

Table des matières

Références

– Hamdy A. Taha, Operations Research, an introduction, Prentice-Hall
– Marc Pirlot, Métaheuristiques pour l'optimisation combinatoire : un aperçu général, Chapitre 1, dans "Optimisation approchée en recherche opérationnelle", sous la direction de Jacques Teghem et Marc Pirlot, Hermes Science.

Première partie

Introduction à la recherche opérationnelle

1 Quelques exemples de modèles mathématiques

Un premier problème

Exemple 1 (Achat de billets d'avion).
- Un homme d'affaires doit effectuer 5 voyages entre Fayetteville (FYV) à Denver (DEN), en partant le lundi de FYV et revenant le mercredi de DEN à FYV.
- Billet aller-retour : $400.
- Réduction de 20 % si un weekend est inclus.
- Aller simple : 75 % du prix aller-retour.

Question
Comment acheter les billets pour les 5 semaines (à prix minimum) ?

Aide à la décision

Problème d'aide à la décision

1. Quelles sont les alternatives possibles ?
2. Quelles sont les restrictions à cette décision ?
3. Quel est l'objectif utilisé pour évaluer les alternatives ?

Restrictions
FYV-DEN le lundi et DEN-FYV le mercredi de la même semaine.

Evaluation des alternatives

Alternatives
- Acheter 5 FYV-DEN-FYV normaux. *5 x $400 = $2000*
- Acheter un FYV-DEN, 4 DEN-FYV-DEN comprenant un weekend et un DEN-FYV. *0.75 x $400 + 4 x 0.8 x $400 + 0.75 x $400 = $1880*
- Acheter un FYV-DEN-FYV pour le lundi de la première semaine et le mercredi de la dernière semaine, et 4 DEN-FYV-DEN comprenant un weekend pour les autres voyages. *5 x 0.8 x* $400 = 1600$

La troisième alternative est la meilleure.

Modèle de recherche opérationnelle

Ingrédients principaux
- *Alternatives* (variables, inconnues du problème).
- *Restrictions* (contraintes).
- *Fonction objectif* à optimiser (minimiser ou maximiser).

Définition 1 (Solution admissible). Une *solution admissible* est un ensemble de valeurs données aux variables qui satisfait toutes les contraintes.

Définition 2 (Solution optimale). Une *solution optimale* est une solution admissible qui optimise la fonction objectif.

Définition 3 (Modèle de recherche opérationnelle). Maximiser ou minimiser *(fonction objectif)* Sujet à *{ contraintes }*

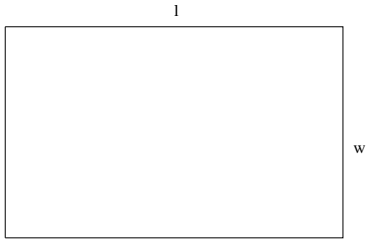
Variables : continues (réelles), entières, booléennes (0/1), ...

Objectif : linéaire / non-linéaire, concave / convexe, ...

Contraintes : linéaire / non-linéaire, concave / convexe, égalités / inégalités, ...

Paramètres : connus avec certitude (modèles déterministes) / incertains (modèles stochastiques)

Exemple 2 (Maximisation de la surface d'un rectangle). Supposons que l'on veut plier un fil de fer de longueur L en rectangle de manière à maximiser la surface du rectangle.



Formulation

$$\begin{array}{ll} \max & A = lw \\ \text{s.t.} & l + w = \frac{L}{2} \end{array}$$

Solution
- $A = \left(\frac{L}{2} - w\right) w = \frac{Lw}{2} - w^2$
- $\frac{dA}{dw} = \frac{L}{2} - 2w = 0$
- Solution optimale : $w = l = \frac{L}{4}$

Méthodes de résolution
- Dans l'exemple, *solution analytique* au problème.
- La plupart des problèmes pratiques sont trop grands ou trop complexes pour être résolus analytiquement.

  Méthodes itératives
  Déplacement de solution en solution pour atteindre l'optimum (*méthodes exactes*) ou une "bonne" solution (*heuristiques*).

- Importance des algorithmes et des solutions informatiques.

2 Tour d'horizon des techniques de recherche opérationnelle

Recherche opérationnelle
La recherche opérationnelle est une technique d'*aide à la décision*.

Etapes pratiques

1. Définition du problème
2. Construction d'un modèle
3. Solution du modèle
4. Validation du modèle
5. Implémentation de la solution

Méthodologie
- Les étapes les plus importantes sont la définition du problème (suppose un *dialogue* avec le décideur) et la construction du modèle (prendre conscience des *hypothèses simplificatrices* et de leur impact).
- La phase de validation doit permettre de *remettre en cause la validité du modèle*.
- Une approche globale nécessite donc un aller-retour constant entre le modèle et les attentes du décideur.

Techniques principales
- Programmation linéaire
- Programmation en nombres entiers
- Optimisation dans les réseaux

- Programmation non linéaire
- "Optimisation" multi-critères
- Programmation dynamique
- Modèles stochastiques
- Simulation

Deuxième partie

Applications de la programmation linéaire

3 Définition, exemples et méthode de résolution

3.1 Notions de bases

Programmation linéaire

Définition 4 (Programme linéaire). Modèle mathématique dans lequel la fonction objectif et les contraintes sont linéaires en les variables.

Applications
Optimisation de l'usage de ressources limitées dans les domaines militaire, industriel, agricole, économique, ...

Existence d'algorithmes très efficaces pour résoudre des problèmes de très grande taille (*simplexe*, points intérieurs)

3.2 Exemples de modèles linéaires

Exemple 3 (Production de peinture). Une société produit de la peinture d'intérieur et d'extérieur à partir de deux produits de base M1 et M2.

Données

	Quantité utilisée par tonne		Quantité disponible par jour
	Extérieure	Intérieure	
M1	6	4	24
M2	1	2	6
Profit par tonne	5	4	

Contraintes supplémentaires
– Demande maximum en peinture d'intérieur : 2 tonnes / jour.
– La production en peinture d'intérieur ne dépasser que d'une tonne celle d'extérieur.

Formulation (Production de peinture)

Alternatives (*variables*, inconnues du problème)

$$x_1 = \text{tonnes de peinture d'extérieur produites par jour}$$

$$x_2 = \text{tonnes de peinture d'intérieur produites par jour}$$

***Fonction objectif* à optimiser**

$$\max z = 5x_1 + 4x_2$$

Restrictions (*contraintes*)

$$\begin{aligned} 6x_1 + 4x_2 &\leq 24 \\ x_1 + 2x_2 &\leq 6 \\ x_2 &\leq 2 \\ x_2 - x_1 &\leq 1 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Solutions et méthodes de résolution
– *Solution admissible :* satisfait toutes les contraintes.

$$x_1 = 3,\ x_2 = 1\ (\Rightarrow z = 19)$$

– Nous voulons trouver la *solution (admissible) optimale*.
– Infinité de solutions admissibles !

Méthodes pour trouver l'optimum
– Méthode graphique
– Simplexe
– (Ellipsoide, points intérieurs)

Exemple 4 (Diet problem). – On désire déterminer la composition, à coût minimal, d'un aliment pour bétail qui est obtenu en mélangeant au plus trois produits bruts : orge et arachide.
– La quantité nécessaire par portion est de 400g.
– L'aliment ainsi fabriqué devra comporter au moins 30% de protéines et au plus 5% de fibres.

Données

	Quantité par gramme d'aliment		Coût
Aliment	Protéines	Fibres	(EUR / kg)
Orge	0.09	0.02	1.5
Arachide	0.60	0.06	4.5

Formulation (Diet problem)

Variables

$$\begin{aligned} x_1 &= \text{grammes d'orge par portion} \\ x_2 &= \text{grammes d'arachide par portion} \end{aligned}$$

Objectif
$\min z = 0.0015x_1 + 0.0045x_2$

Contraintes

Quantité totale : $x_1 + x_2 \geq 400$

Protéines : $0.09x_1 + 0.6x_2 \geq 0.3(x_1 + x_2)$

Fibres : $0.02x_1 + 0.06x_2 \leq 0.05(x_1 + x_2)$

Non-négativité : $x_1, x_2 \geq 0$

3.3 Forme standard et forme canonique d'un programme linéaire

Forme standard

Définition 5 (Forme standard). Un programme linéaire est sous *forme standard* lorsque toutes ses contraintes sont des *égalités* et toutes ses variables sont non-négatives.

Représentation matricielle

$$\begin{aligned} \max \quad & c^T x \\ \text{s.c.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

n variables, m contraintes, $m < n$, $c, x \in \mathbb{R}^n$, $\quad b \in \mathbb{R}^m$, $\quad A \in \mathbb{R}^{m \times n}$.

Forme canonique

Définition 6 (Forme canonique). Un programme linéaire est sous *forme canonique* lorsque toutes ses contraintes sont des *inégalités* et toutes ses variables sont non-négatives.

Représentation matricielle

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

n variables, m contraintes, $c, x \in \mathbb{R}^n$, $\quad b \in \mathbb{R}^m$, $\quad A \in \mathbb{R}^{m \times n}$.

Théorème 1 (Equivalence des formes standard et canonique). Tout programme linéaire peut s'écrire sous forme standard et sous forme canonique.

Démonstration.
– Une containte d'inégalité $a^T x \leq b$ peut être transformée en égalité par l'introduction d'une *variable d'écart* :

$$\begin{aligned} a^T x + s &= b, \\ s &\geq 0. \end{aligned}$$

– Une contrainte d'égalité $a^T x = b$ peut être remplacée par deux inégalités :

$$\begin{aligned} a^T x &\leq b \\ -a^T x &\leq -b \end{aligned}$$

– $a^T x \geq b \Leftrightarrow -a^T x \leq -b$.
– $\min c^T x = -\max -c^T x$.
– Variable x non restreinte : substitution par deux variables (partie positive et négative)

$$\begin{aligned} x &= x^+ - x^- \\ x^+, x^- &\geq 0. \end{aligned}$$

Il existe toujours une solution optimale telle que $x^+ = 0$ ou $x^- = 0$.

□

Forme standard du problème de production de peinture

$$\begin{aligned} \max z = 5x_1 + 4x_2 & \\ s.c. 6x_1 + 4x_2 &\leq 24 \\ x_1 + 2x_2 &\leq 6 \\ x_2 &\leq 2 \\ x_2 - x_1 &\leq 1 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Forme standard

$$\begin{array}{rrrrrrrrrr} \max z = & 5x_1 & +4x_2 & & & & & & & \\ \text{s.c.} & 6x_1 & +4x_2 & +s_1 & & & & & = & 24 \\ & x_1 & +2x_2 & & +s_2 & & & & = & 6 \\ & & x_2 & & & +s_3 & & & = & 2 \\ & -x_1 & +x_2 & & & & +s_4 & & = & 1 \\ & x_1, & x_2, & s_1, & s_2, & s_3, & s_4 & & \geq & 0 \end{array}$$

Forme matricielle

$$\begin{aligned} \max \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

$$c = \begin{pmatrix} 5 \\ 4 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, x = \begin{pmatrix} x_1 \\ x_2 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \end{pmatrix}, A = \begin{pmatrix} 6 & 4 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, b = \begin{pmatrix} 24 \\ 6 \\ 2 \\ 1 \end{pmatrix}$$

Variables pouvant prendre des valeurs négatives

Exemple 5 (Vente de hamburgers).

- Un fast-food vend des hamburgers et des cheeseburgers. Un hamburger utilise 125 g. de viande alors qu'un cheeseburger n'en utilise que 100 g.
- Le fast-food démarre chaque journée avec 10 kg de viande mais peut commander de la viande supplémentaire avec un coût additionnel de 2 EUR par kg pour la livraison.
- Le profit est de 0.02 EUR pour un hamburger et 0.015 EUR pour un cheeseburger.
- La demande ne dépasse pas 900 sandwiches / jour, et les surplus de viande sont donnés au Restos du Coeur.

Combien le fast-food doit-il produire de sandwiches de chaque type par jour ?

Variables

$x_1 =$ nombre de hamburgers / jour $x_2 =$ nombre de cheeseburgers / jour

Contraintes

- Commande de viande supplémentaire :

$$125x_1 + 100x_2 + x_3 = 10000, \qquad x_3 \text{non restreint}$$

- Le coût pour la viande supplémentaire apparaît seulement si $x_3 < 0$.

– Substitution de x3 par 2 variables non-négatives :

$$x_3 = x_3^+ - x_3^-,\ x_3^+, x_3^- \geq 0$$
$$125x_1 + 100x_2 + x_3^+ - x_3^- = 10000$$

– Borne supérieure sur les ventes : $x_1 + x_2 \leq 900$.

Modèle complet

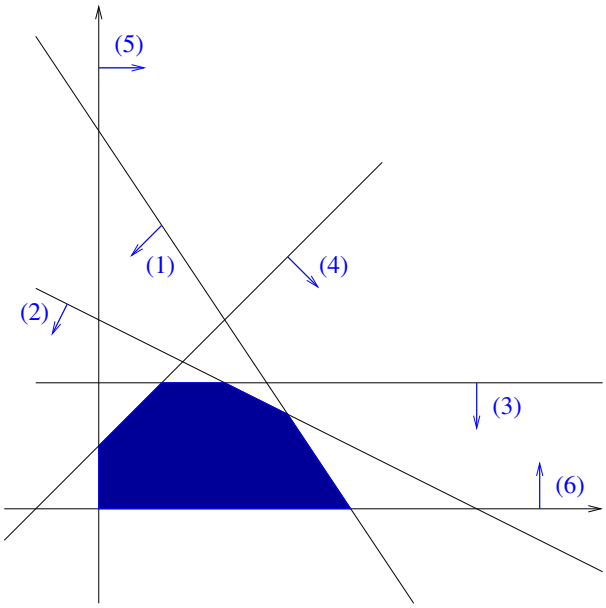
$$\begin{array}{rll} \max z = & 0.02x_1 + 0.015x_2 - 0.002x_3^- & \\ \text{s.c.} & 125x_1 + 100x_2 + x_3^+ - x_3^- & = 10000 \\ & x_1 + x_2 & \leq 900 \\ & x_1, x_2, x_3^+, x_3^- & \geq 0 \end{array}$$

Remarque : Il existe une solution optimale telle que $x_3^+ = 0$ ou $x_3^- = 0$.

3.4 Résolution de programmes linéaires

3.4.1 Résolution graphique

Représentation graphique



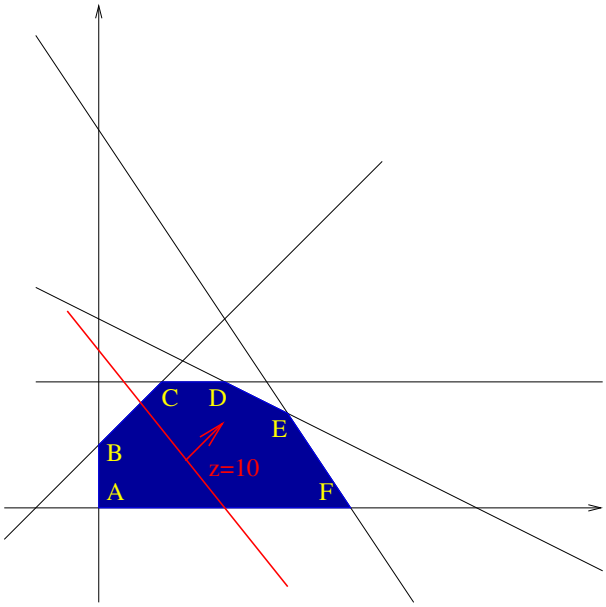
Production de peinture

$$\max z = 5x_1 + 4x_2$$

sous les contraintes :

$$\begin{array}{rcll} 6x_1 + 4x_2 & \leq & 24 & (1) \\ x_1 + 2x_2 & \leq & 6 & (2) \\ x_2 & \leq & 2 & (3) \\ x_2 - x_1 & \leq & 1 & (4) \\ x_1 & \geq & 0 & (5) \\ x_2 & \geq & 0 & (6) \end{array}$$

Géométrie des solutions

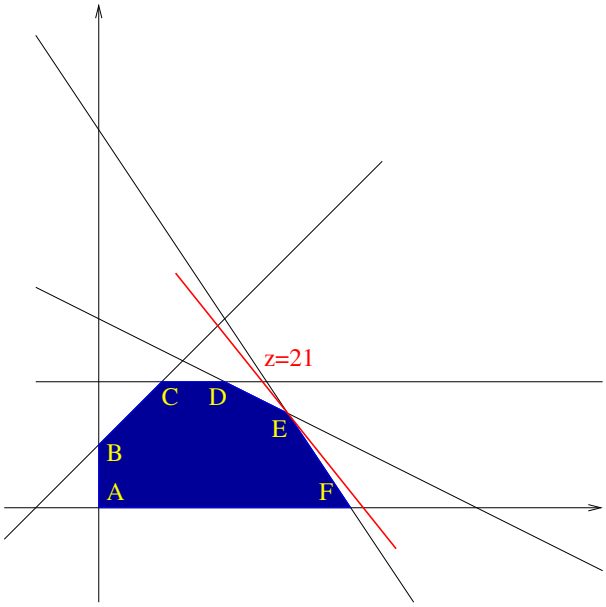


Ensemble des solutions admissibles
Polyèdre (ABCDEF)

Courbes de niveaux de l'objectif
Ensemble de solutions ayant un profit (valeur de l'objectif) donné : intersection entre une droite et le polyèdre.

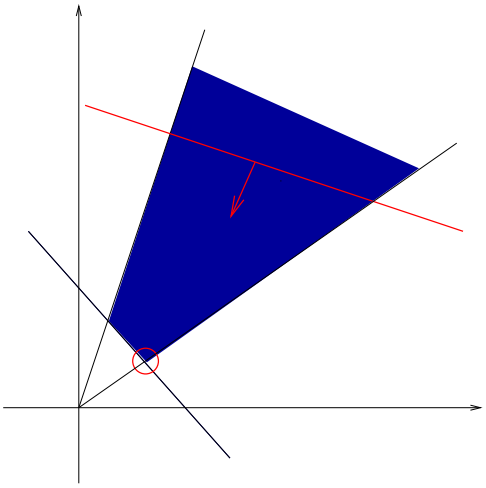
Amélioration de la solution
Recherche d'une direction dans laquelle le profit z augmente.

Résolution graphique (Production de peinture)



Recherche de la solution optimale
- La droite mobile doit garder une intersection avec l'ensemble des solutions admissibles.
- Solution optimale : $x_1 = 3$, $x_2 = 1.5$ (E)
- La solution optimale est un *sommet du polyèdre*.
- Cette observation est la base de l'algorithme du simplexe.

Résolution graphique (Diet problem)



Diet problem

$$\min z = 0.0015x_1 + 0.0045x_2$$

sous les contraintes

$$\begin{aligned} x_1 + x_2 &\geq 400 \\ 0.21x_1 - 0.30x_2 &\leq 0 \\ 0.03x_1 - 0.01x_2 &\geq 0 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

Solution optimale

$$x_1 = \frac{4000}{17} \simeq 235.3 \quad x_2 = \frac{2800}{17} \simeq 164.7 \quad z = \frac{186}{170} \simeq 1.094$$

3.4.2 La méthode du simplexe

Idées de base

- Solution optimale : *sommet (point extrême)*.
- Idée fondamentale du simplexe : déplacement de sommet en sommet adjacent de manière à améliorer la fonction objectif.
- Transformation des inégalités en égalités : *forme standard* du programme linéaire - système de m équations à n inconnues ($m < n$).
- Identification algébrique des sommets : correspondance avec les bases d'un système d'équations.

Solutions de base

- Système de m équations linéaires à n inconnues ($m < n$) : infinité de solutions.
- Si on fixe à zéro $n - m$ variables : système de m équations à m inconnues possédant une solution unique (si la matrice est inversible). C'est une *solution de base*.

Définition 7 (Solution de base). Une *solution de base* d'un programme linéaire est la solution unique du système de m équations à m inconnues obtenu en fixant à zéro $n - m$ variables (pourvu que la matrice du système soit inversible).
Les variables fixées à zéro sont appelées *variables hors base* et les autres *variables en base*.

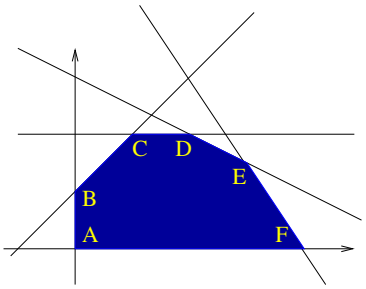
Exemple 6 (Production de peinture). Prenons $B = \{s_1, s_2, s_3, s_4\}$.

$$\begin{aligned} z &= 0 + 5x_1 + 4x_2 \\ s_1 &= 24 - 6x_1 - 4x_2 \\ s_2 &= 6 - x_1 - 2x_2 \\ s_3 &= 2 - x_2 \\ s_4 &= 1 + x_1 - x_2 \end{aligned}$$

Si $x_1 = x_2 = 0$, alors $s_1 = 24,\ s_2 = 6,\ s_3 = 2,\ s_4 = 1$. Toutes ces valeurs sont non-négatives et la solution est réalisable.

Définition 8 (Solution de base réalisable). Une solution de base telle que toutes les variables prennent des valeurs non-négatives est appelée *solution de base réalisable*.

Géométrie des solutions de base



- Prenons $B = \{s_1, s_2, s_3, s_4\} \Rightarrow x_1 = x_2 = 0,\ s_1 = 24,\ s_2 = 6,\ s_3 = 2,\ s_4 = 1$.
- Cette solution de base réalisable correspond au sommet $(0, 0)$.

Base	Solution	Objectif	Sommet
$\{s_1, s_2, s_3, s_4\}$	$(0, 0)$	0	A
$\{x_1, s_2, s_3, s_4\}$	$(4, 0)$	20	F
$\{s_1, x_1, s_3, s_4\}$	$(6, 0)$	–	Non réalisable
$\{x_1, x_2, s_3, s_4\}$	$(3, 1.5)$	21	E

Théorème 2. Toute solution de base réalisable correspond à un sommet du polyèdre.

Détermination de la solution de base optimale

- Nombre maximum de solutions de base : $\frac{n!}{m!(n-m)!}$
- Algorithme "bête et méchant" : énumération de toutes les bases.
- Méthode du simplexe : partir d'une solution de base admissible et passer à une solution de base voisine qui améliore la valeur de l'objectif.
- Solution voisine : changement d'une variable en base.
- 3 etapes :
  1. Détermination de la variable entrante.
  2. Détermination de la variable sortante.
  3. Pivotage.

L'algorithme du simplexe

Variable entrante

$$\begin{aligned} z &= 0 + 5x_1 + 4x_2 \\ s_1 &= 24 - 6x_1 - 4x_2 \\ s_2 &= 6 - x_1 - 2x_2 \\ s_3 &= 2 - x_2 \\ s_4 &= 1 + x_1 - x_2 \end{aligned}$$

- Si x_1 (ou x_2) augmente (*entre en base*), la valeur de la fonction objectif z augmente.
- Quelle est la valeur maximale de x_1 ?

– Contraintes : les autres variables doivent rester positives.

Variable sortante

$$\begin{array}{lrlll} s_1 = & 24 - 6x_1 \geq 0 & \rightarrow x_1 \leq 4 & & \\ s_2 = & 6 - x_1 \geq 0 & \rightarrow x_1 \leq 6 & & \\ s_3 = & 2 \geq 0 & \rightarrow 2 \geq 0 & toujours! & \Rightarrow x_1 \leq 4 \\ s_4 = & 1 + x_1 \geq 0 & \rightarrow x_1 \geq -1 & toujours! & \end{array}$$

Pivotage

– Si $x_1 = 4$, alors $s_1 = 0$.
– x_1 *entre en base*, s_1 *sort de la base*.
– Substitution :

$$x_1 = 4 - \frac{1}{6}s_1 - \frac{2}{3}x_2$$

– Nouveau système :

$$\begin{array}{rrrr} z = & 20 & -\frac{5}{6}s_1 & +\frac{2}{3}x_2 \\ x_1 = & 4 & -\frac{1}{6}s_1 & -\frac{2}{3}x_2 \\ s_2 = & 2 & +\frac{1}{6}s_1 & -\frac{4}{3}x_2 \\ s_3 = & 2 & & -x_2 \\ s_4 = & 5 & -\frac{1}{6}s_1 & -\frac{5}{3}x_2 \end{array}$$

Equations du simplexe

– $B =$ indices des variables en base, $N =$ indices des variables hors base.
– Notation :

$$z = \overline{z} + \sum_{k \in N} \overline{c_k} x_k$$
$$x_l = \overline{b_l} - \sum_{k \in N} \overline{a_{lk}} x_k \quad l \in B$$

– c_k : *profit marginal* ou *coût réduit*.

Règles de pivotage

Variable entrante

Choisir la variable k hors base avec le profit marginal maximum ($\max z$) ou le coût réduit minimum ($\min z$).

$$\begin{array}{rcl} \max z & \rightarrow & k = \arg\max_{i \in N} \overline{c_i} \\ \min z & \rightarrow & k = \arg\min_{i \in N} \overline{c_i} \end{array}$$

Si $\overline{c_k} \leq 0$ ($\max$) ou $\overline{c_k} \geq 0$ ($\min$) pour tout $k \in N$, *solution optimale*, STOP.

$$\begin{array}{rrrr} z = & 0 & \boxed{+5x_1} & +4x_2 \\ s_1 = & 24 & -6x_1 & -4x_2 \\ s_2 = & 6 & -x_1 & -2x_2 \\ s_3 = & 2 & & -x_2 \\ s_4 = & 1 & +x_1 & -x_2 \end{array}$$

Variable sortante

Choisir la variable l en base telle que

$$l = \arg \min_{j \in B : \overline{a_{jk}} > 0} \frac{\overline{b_j}}{\overline{a_{jk}}}$$

Si $\overline{a_{lk}} \leq 0$ pour tout $l \in B$, *problème non borné*, STOP.

$$\begin{aligned} z &= 0 + 5x_1 + 4x_2 \\ s_1 &= 24 \boxed{-6x_1} - 4x_2 \\ s_2 &= 6 - x_1 - 2x_2 \\ s_3 &= 2 - x_2 \\ s_4 &= 1 + x_1 - x_2 \end{aligned}$$

Pivotage

$$\overline{a_{ij}}' = \begin{cases} \dfrac{\overline{a_{lj}}}{\overline{a_{lk}}} & i = l \\ \overline{a_{ij}} - \dfrac{\overline{a_{ik}}\overline{a_{lj}}}{\overline{a_{lk}}} & i \neq l \end{cases}$$

$$\overline{b_i}' = \begin{cases} \dfrac{\overline{b_l}}{\overline{a_{lk}}} & i = l \\ \overline{b_i} - \dfrac{\overline{a_{ik}}\overline{b_l}}{\overline{a_{lk}}} & i \neq l \end{cases}$$

$$\begin{aligned} z &= 0 + 5x_1 + 4x_2 \\ s_1 &= 24 \boxed{-6x_1} - 4x_2 \\ s_2 &= 6 - x_1 - 2x_2 \\ s_3 &= 2 - x_2 \\ s_4 &= 1 + x_1 - x_2 \end{aligned}$$

$$\begin{aligned} z &= 20 - \tfrac{5}{6}s_1 + \tfrac{2}{3}x_2 \\ x_1 &= 4 - \tfrac{1}{6}s_1 - \tfrac{2}{3}x_2 \\ s_2 &= 2 + \tfrac{1}{6}s_1 \boxed{-\tfrac{4}{3}x_2} \\ s_3 &= 2 - x_2 \\ s_4 &= 5 - \tfrac{1}{6}s_1 - \tfrac{5}{3}x_2 \end{aligned}$$

$$\begin{aligned} z &= 21 - \tfrac{3}{4}s_1 - \tfrac{1}{2}s_2 \\ x_1 &= 3 - \tfrac{1}{4}s_1 + \tfrac{1}{2}s_2 \\ x_2 &= \tfrac{3}{2} + \tfrac{1}{8}s_1 - \tfrac{3}{4}s_2 \\ s_3 &= \tfrac{1}{2} - \tfrac{1}{8}s_1 + \tfrac{3}{4}s_2 \\ s_4 &= \tfrac{5}{2} - \tfrac{3}{8}s_1 + \tfrac{5}{4}s_2 \end{aligned}$$

Présentation en tableau
Présentation compacte pour effectuer les calculs sans répéter les systèmes d'équations.

Itération 1

Var. en base	z	x_1	x_2	s_1	s_2	s_3	s_4	Solution
z	-1	5	4	0	0	0	0	0
s_1	0	6	4	1	0	0	0	24
s_2	0	1	2	0	1	0	0	6
s_3	0	0	1	0	0	1	0	2
s_4	0	-1	1	0	0	0	1	1

Itération 2

Var. en base	z	x_1	x_2	s_1	s_2	s_3	s_4	Solution
z	-1	0	$\frac{2}{3}$	$-\frac{5}{6}$	0	0	0	-20
x_1	0	1	$\frac{2}{3}$	$\frac{1}{6}$	0	0	0	4
s_2	0	0	$\frac{4}{3}$	$-\frac{1}{6}$	1	0	0	2
s_3	0	0	1	0	0	1	0	2
s_4	0	0	$\frac{5}{3}$	$\frac{1}{6}$	0	0	1	5

Itération 3

Var. en base	z	x_1	x_2	s_1	s_2	s_3	s_4	Solution
z	-1	0	0	$-\frac{3}{4}$	$-\frac{1}{2}$	0	0	-21
x_1	0	1	0	$\frac{1}{4}$	$-\frac{1}{2}$	0	0	3
x_2	0	0	1	$-\frac{1}{8}$	$\frac{3}{4}$	0	0	$\frac{3}{2}$
s_3	0	0	0	$\frac{1}{8}$	$-\frac{3}{4}$	1	0	$\frac{1}{2}$
s_4	0	0	0	$\frac{3}{8}$	$-\frac{5}{4}$	0	1	$\frac{5}{2}$

3.4.3 La méthode des deux phases

Solution initiale artificielle
- Une solution de base admissible n'est pas toujours connue a priori.
- Certains problèmes n'admettent pas de solution admissible, donc il est impossible de trouver une base de départ.
- La méthode des *deux phases* va permettre de déterminer une base admissible ou prouver que le problème est impossible.

Exemple 7 (Méthode des 2 phases).

$$\begin{array}{lllllll} \min & z = & 4x_1 & +x_2 & & & \\ \text{s.c.} & & 3x_1 & +x_2 & = & 3 \\ & & 4x_1 & +3x_2 & \geq & 6 \\ & & x_1 & +2x_2 & \leq & 4 \\ & & x_1, & x_2 & \geq 0 & & \end{array}$$

Introduction des variables d'écart

$$\begin{array}{lllllllll} \min & z = & 4x_1 & +x_2 & & & & & \\ \text{s.c.} & & 3x_1 & +x_2 & & & & = & 3 \\ & & 4x_1 & +3x_2 & -x_3 & & & = & 6 \\ & & x_1 & +2x_2 & & +x_4 & & = & 4 \\ & & x_1, & x_2, & x_3, & x_4 & \geq 0 & & \end{array}$$

– Pas de base admissible "triviale".
– On voudrait voir apparaître une *matrice identité*.
– Introduction de *variables artificielles*.

Introduction des variables artificielles

$$
\begin{array}{llrrrrrrl}
\min & z = & 4x_1 & +x_2 & & & & & \\
\min & r = & & & & R_1 & +R_2 & & \\
\min & r = & -7x_1 & -4x_2 & +x_3 & & & & +9 \\
\text{s.c.} & & 3x_1 & +x_2 & & +R_1 & & = & 3 \\
& & 4x_1 & +3x_2 & -x_3 & & +R_2 & = & 6 \\
& & x_1 & +2x_2 & & & +x_4 & = & 4 \\
& & x_1, & x_2, & x_3, & R_1, & R_2, & x_4 & \geq 0
\end{array}
$$

– R_1, R_2 et x_4 peuvent être utilisées comme base de départ admissible.
– Base pour le système de départ si $R_1 = R_2 = 0$ (hors base).
– Réécrire l'objectif en fonction des variables hors base !

3.4.4 Cas particuliers

Solutions optimales multiples

– Si la fonction objectif est parallèle à une contrainte active pour la solution optimale, la même valeur de l'objectif peut être prise par plusieurs solutions admissibles.
– Il y a une *infinité* de solutions optimales dans ce cas (toutes les combinaisons convexes de sommets optimaux).
– Cela se traduit par un *profit marginal nul* pour une ou plusieurs variables hors base.

Exemple 8 (Solutions optimales multiples).

$$
\begin{array}{llrrrr}
\max & z = & 2x_1 & +4x_2 & & \\
\text{s.c.} & & x_1 & +2x_2 & \leq & 5 \\
& & x_1 & +x_2 & \leq & 4 \\
& & x_1, & x_2 & \geq 0 &
\end{array}
$$

Var. en base	z	x_1	x_2	s_1	s_2	Solution
z	-1	2	4	0	0	0
s_1	0	1	2	1	0	5
s_2	0	1	1	0	1	4
z	-1	0	0	-2	0	-10
x_2	0	$\frac{1}{2}$	1	$\frac{1}{2}$	0	$\frac{5}{2}$
s_2	0	$\frac{1}{2}$	0	$-\frac{1}{2}$	1	$\frac{3}{2}$
z	-1	0	0	-2	0	-10
x_2	0	0	1	1	-1	1
x_1	0	1	0	-1	2	3

Solution optimale :

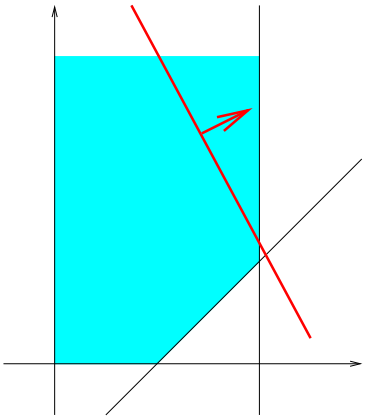
$$
\begin{aligned}
x_1 &= 0\alpha + 3(1-\alpha) &&= 3 - 3\alpha \\
x_2 &= \tfrac{5}{2}\alpha + 1(1-\alpha) &&= 1 + \frac{3}{2}\alpha \quad (0 \leq \alpha \leq 1)
\end{aligned}
$$

Problèmes non bornés

– Certains problèmes sont *non bornés* dans une direction donnée.
– Si cette direction est une direction d'amélioration de la fonction objectif, celle-ci peut prendre une valeur arbitrairement grande !

Exemple 9 (Problèmes non bornés).

$$\begin{array}{lrrrrr} \max & z = & 2x_1 & +x_2 & & \\ \text{s.c.} & & x_1 & -x_2 & \leq & 1 \\ & & 2x_1 & & \leq & 4 \\ & & x_1, & x_2 & \geq 0 & \end{array}$$



Var. en base	z	x_1	x_2	s_1	s_2	Solution
z	-1	2	1	0	0	0
s_1	0	1	-1	1	0	1
s_2	0	2	0	0	1	4

– Tous les coefficients (sauf le profit maginal) dans la colonne de x_2 sont négatifs ou nuls.
– Cela signifie que toutes les contraintes de non-négativité sont satisfaites quelle que soit la valeur de x_2.
– L'objectif peut donc augmenter indéfiniment.

Problèmes impossibles
– Le système de contraintes peut n'avoir aucune solution.
– Généralement, provient d'une mauvaise formulation du problème.

Exemple 10 (Problèmes impossibles).

$$\begin{array}{lrrrrr} \max & z = & 3x_1 & +2x_2 & & \\ \text{s.c.} & & 2x_1 & +x_2 & \leq & 2 \\ & & 3x_1 & +4x_2 & \geq & 12 \\ & & x_1, & x_2 & \geq 0 & \end{array}$$

4 Dualité

4.1 Le problème dual

Problème primal et problème dual

Problème primal

$$\begin{array}{ll} \max & c^T x \\ \text{s.c.} & Ax = b \\ & x \geq 0 \end{array}$$

n variables, m contraintes, $m < n$, $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$.

Problème dual

$$\begin{array}{ll} \min & b^T y \\ \text{s.c.} & A^T y \geq c \\ & (y \text{ non restreint}) \end{array}$$

m variables, n contraintes, $m < n$, $c \in \mathbb{R}^n$, $b, y \in \mathbb{R}^m$, $A \in \mathbb{R}^{m \times n}$.

Exemple 11 (Problème primal et dual - forme standard).

Problème primal :

$$\begin{array}{llrrrl} \max & z = & x_1 & +x_2 & & \\ \text{s.c.} & & 2x_1 & +x_2 & = 5 & (y_1) \\ & & 3x_1 & -x_2 & = 6 & (y_2) \\ & & x_1, & x_2 & \geq 0 & \end{array}$$

Problème dual :

$$\begin{array}{llrrrl} \min & w = & 5y_1 & +6y_2 & & \\ \text{s.c} & & 2y_1 & +3y_2 & \geq 1 & (x_1) \\ & & y_1 & -y_2 & \geq 1 & (x_2) \end{array}$$

Propriétés et règles de construction du dual

Théorème 3. Le problème dual du problème dual est le problème primal.

Règles de construction

Problème max	Problème min
Contrainte	Variable
$\leq$	≥ 0
=	non restreinte
Variable	Contrainte
≥ 0	$\geq$
non restreinte	=

Exemple 12 (Problème primal et dual - forme générale).

Problème primal :

$$\begin{array}{llrrrrl} \max & z = & 5x_1 & +12x_2 & +4x_3 & & \\ \text{s.c.} & & x_1 & +2x_2 & +x_3 & \leq 10 & (y_1) \\ & & 2x_1 & -x_2 & +3x_3 & = 8 & (y_2) \\ & & x_1, & x_2, & x_3 & \geq 0 & \end{array}$$

Problème dual :

$$\begin{array}{lllllll} \min & w = & 10y_1 & +8y_2 & & & \\ \text{s.c} & & y_1 & +2y_2 & \geq 5 & (x_1) \\ & & 2y_1 & -y_2 & \geq 12 & (x_2) \\ & & y_1 & +3y_2 & \geq 4 & (x_3) \\ & & y_1 & & \geq 0 & \end{array}$$

4.2 Relations primal/dual

Théorème 4 (Dualité faible). Considérons la paire primale-duale :

$$\begin{array}{ll} \max & c^T x \\ \text{s.c.} & Ax = b \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \min & b^T y \\ \text{s.c.} & A^T y \geq c \end{array}$$

– Si x est une solution admissible du primal et y une solution admissible du dual, alors

$$c^T x \leq b^T y$$

– S'il y a égalité, alors x est une solution optimale du primal et y une solution optimale du dual.

Théorème 5 (Dualité forte). Considérons la paire primale-duale :

$$\begin{array}{ll} \max & c^T x \\ \text{s.c.} & Ax = b \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \min & b^T y \\ \text{s.c.} & A^T y \geq c \end{array}$$

– Si le primal et le dual admettent tous les deux une solution admissible, ils ont tous deux une solution optimale finie et la même valeur objectif optimale.
– Si le primal (dual) est non borné, le dual (primal) n'admet pas de solution admissible.

Théorème 6 (Complémentarité). Considérons la paire primale-duale :

$$\begin{array}{ll} \max & c^T x \\ \text{s.c.} & Ax = b \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \min & b^T y \\ \text{s.c.} & A^T y \geq c \end{array}$$

Si x est une solution optimale du primal et y une solution optimale du dual, alors

$$x_i(a_i^T y - c_i) = 0.$$

où a_i est la i-ème colonne de A.
En d'autres termes :

$$\begin{aligned} x_i > 0 &\Rightarrow a_i^T y = c_i, \\ a_i^T y > c_i &\Rightarrow x_i = 0. \end{aligned}$$

Exemple 13 (Résolution du dual par les règles de complémentarité).
Primal (P) :

$$\begin{array}{lllllll} \max & z = & 5x_1 & +12x_2 & +4x_3 & & \\ \text{s.c.} & & x_1 & +2x_2 & +x_3 & \leq 10 & (y_1) \\ & & 2x_1 & -x_2 & +3x_3 & = 8 & (y_2) \\ & & x_1, & x_2, & x_3 & \geq 0 & \end{array}$$

Dual (D) :

$$\begin{array}{lllll} \min & w = & 10y_1 & +8y_2 & \\ \text{s.c} & & y_1 & +2y_2 & \geq 5 \quad (x_1) \\ & & 2y_1 & -y_2 & \geq 12 \quad (x_2) \\ & & y_1 & +3y_2 & \geq 4 \quad (x_3) \\ & & y_1 & & \geq 0 \end{array}$$

Solution optimale de (P) :

$$(x_1, x_2, x_3) = \left(\frac{26}{5}, \frac{12}{5}, 0\right)$$
$$z = \frac{274}{5}$$

$$\begin{aligned} x_1 > 0 &\Rightarrow y_1 + 2y_2 = 5 \\ x_2 > 0 &\Rightarrow 2y_1 - y_2 = 12 \end{aligned}$$

Solution optimale de (D) :

$$(y_1, y_2) = \left(\frac{29}{5}, -\frac{2}{5}\right)$$
$$w = \frac{274}{5}$$

4.3 Interprétation économique de la dualité

– La *forme canonique* d'un programme linéaire peut être interprétée comme un problème d'*allocation de ressources*.
– Paire primale-duale :

$$\begin{array}{ll} \max & c^T x \\ \text{s.c.} & Ax \leq b \\ & x \geq 0 \end{array}$$

$$\begin{array}{ll} \min & b^T y \\ \text{s.c.} & A^T y \geq c \\ & y \geq 0 \end{array}$$

– Données :

  c_j **:** profit par unité d'activité j.

  b_i **:** disponibilité de la ressource i.

  a_{ij} **:** consommation de la ressource i par unité d'activité j.

– Variables :

  x_j **:** niveau de l'activité j.

  y_i **:** *valeur d'une unité de la ressource* i.

Interprétation de la dualité faible

$$z \leq w : \qquad \text{profit } \leq \text{ valeur des ressources}$$

Interprétation de la dualité forte

Le profit maximal est atteint si les ressources ont été exploitées complètement, i.e. jusqu'à épuisement de leur valeur.

Exemple 14 (Dualité dans le problème de production de peinture).

$$\begin{array}{rrrrcl} \max z = & 5x_1 & +4x_2 & & & \\ \text{s.c} & 6x_1 & +4x_2 & \leq & 24 \\ & x_1 & +2x_2 & \leq & 6 \\ & & x_2 & \leq & 2 \\ & -x_1 & +x_2 & \leq & 1 \\ & x_1, & x_2 & \geq & 0 \end{array}$$

$$\begin{array}{rrrrrl} \min w = & 24y_1 & +6y_2 & +2y_3 & +y_4 & \\ & 6y_1 & +y_2 & & -y_4 & \geq 5 \\ & 4y_1 & +2y_2 & +y_3 & +y_4 & \geq 4 \\ & y_1, & y_2, & y_3, & y_4 & \geq 0 \end{array}$$

$$x_1 = 3,\ x_2 = 1.5,\ z = 21$$
$$y_1 = 0.75,\ y_2 = 0.5,\ y_3 = y_4 = 0,\ w = 21$$

– Le profit augmente de 0.75 par augmentation d'une tonne de M1 et de 0.5 par tonne de M2. (Localement. Dans quelles limites ? Voir analyse de sensibilité)

– Les "ressources" 3 et 4 sont abondantes, augmenter ces ressources n'apporte aucun profit supplémentaire.

5 Solveurs et langages de modélisation

Exemple 15 (Production de jouets).
- Une société de jouets produit des trains, des camions et des voitures, en utilisant 3 machines.
- Les disponibilités quotidiennes des 3 machines sont 430, 460 et 420 minutes, et les profits par train, camion et voiture sont respectivement EUR 3, EUR 2 et EUR 5.
- Les temps nécessaires sur chaque machine sont :

Machine	Train	Camion	Voiture
1	1	2	1
2	3	0	2
3	1	4	0

Primal

$$\begin{array}{rcccc} \max z = & 3x_1 & +2x_2 & +5x_3 & \\ \text{s.c.} & x_1 & +2x_2 & +x_3 & \leq 430 \\ & 3x_1 & & +2x_3 & \leq 460 \\ & x_1 & +4x_2 & & \leq 420 \\ & x_1, & x_2, & x_3 & \geq 0 \end{array}$$

$$\begin{aligned} x_1 &= 0 \\ x_2 &= 100 \\ x_3 &= 230 \\ z &= 1350 \end{aligned}$$

Dual

$$\begin{array}{rcccc} \min w = & 430y_1 & +460y_2 & +420y_3 & \\ \text{s.c.} & y_1 & +3y_2 & +y_3 & \geq 3 \\ & 2y_1 & & +4y_3 & \geq 2 \\ & y_1 & +2y_2 & & \geq 5 \\ & y_1, & y_2, & y_3 & \geq 0 \end{array}$$

$$\begin{aligned} y_1 &= 1 \\ y_2 &= 2 \\ y_3 &= 0 \end{aligned}$$

Var. en base	z	x_1	x_2	x_3	s_1	s_2	s_3	Solution
z	-1	-4	0	0	-1	-2	0	-1350
x_2	0	$-\frac{1}{4}$	1	0	$\frac{1}{2}$	$-\frac{1}{4}$	0	100
x_3	0	$\frac{3}{2}$	0	1	0	$\frac{1}{2}$	0	230
s_3	0	2	0	0	-2	1	1	20

Base : $B = \{x_2, x_3, s_3\}$.

Solveurs

Logiciels pour résoudre des programmes linéaires :
- Indépendants :
  - Commerciaux : CPLEX (www.ilog.com), XPRESS-MP (www.dash.co.uk), ...
  - Gratuits : PCx, lpsolve, glpk, ...
- Tableurs : La plupart des tableurs intègrent un outil de résolution de programmes linéaires (Excel, Gnumeric, ...)
- Langages de modélisation (ampl, GNU MathProg, mpl, OPL studio, mosel, ...) : langages de haut niveau permettant la séparation modèle/données, se chargeant de l'interface avec un solveur.

```
NAME          toys
ROWS
 L  R0001
 L  R0002
 L  R0003
 N  R0004
COLUMNS
    C0001     R0001     1
    C0001     R0002     3
    C0001     R0003     1
    C0001     R0004     3
    C0002     R0001     2
    C0002     R0003     4
    C0002     R0004     2
    C0003     R0001     1
    C0003     R0002     2
    C0003     R0004     5
RHS
    B         R0001     430
    B         R0002     460
    B         R0003     420
ENDATA
```

FIGURE 1 – Exemple de production de jouets au format MPS

Solveurs indépendants

Avantages
- Puissance, efficacité
- Intégrables dans des applications via des librairies

Désavantages
- Formats de fichiers (MPS)
- Pas de séparation modèle / données
- Ré-utilisation difficile des modèles

Solveurs intégrés aux tableurs

Avantages
- Disponibles sur (quasi) tous les ordinateurs
- Interface facile d'utilisation
- Présentation des données / résultats

Désavantages
- Difficulté d'implémenter de grands modèles
- Séparation modèle / données difficile
- Solveurs moins efficaces (en général)

Langages de modélisation

Avantages
- Séparation modèle / données
- Ré-utilisabilité des modèles
- Indépendance modèle / solveur

```
#
# Data definition
#

set Toys;
param nMachines;
set Machines := 1..nMachines;

param profit {Toys};
param time {Machines,Toys};
param avail {Machines};

#
# Variables
#

var prod {Toys} >= 0;

#
# Objective
#

maximize total_profit:
sum{t in Toys} profit[t]*prod[t];

#
# Constraints
#

subject to machine_usage {m in Machines}:
sum{t in Toys} time[m,t] * prod[t] <= avail[m];
```

FIGURE 2 – Exemple de production de jouets au format AMPL (modèle)

Désavantages
- Apprentissage du langage
- Prix des versions commerciales
- Limitation en taille des versions d'essai gratuites
- Moindre efficacité (actuellement) des solveurs gratuits

```
#
# Data
#

data;

set Toys := Trains, Trucks, Cars ;
param nMachines := 3;

param profit :=
Trains  3
Trucks  2
Cars  5 ;

param time :  Trains Trucks Cars :=
1 1 2 1
2 3 0 2
3 1 4 0 ;

param avail :=
1  430
2 460
3 420 ;

end;
```

FIGURE 3 – Exemple de production de jouets au format AMPL (données)

Troisième partie

Programmation en nombres entiers et optimisation combinatoire

6 Définitions et exemples

Programmation en nombres entiers

– Programmes linéaires dans lesquels certaines (ou toutes les) variables sont restreintes à des valeurs entières.
– Si seulement une partie des variables doivent être entières, on parle de *programme mixte* (MILP).
– *Optimisation combinatoire :* choix d'une solution optimale parmi un ensemble fini d'alternatives. Peuvent généralement se formuler comme des programmes en nombres entiers.
– Problèmes très difficiles en pratique, mais solveurs de plus en plus performants.

Exemple 16 (Sélection de projets). 5 projets doivent être évalués sur 3 ans. Etant donné le coût de chaque projet pour chaque année et le profit obtenu par l'éxécution d'un projet, décider quels projets éxécuter sans dépasser le budget disponible pour chaque année.

	Coût par année			Profit
Projet	1	2	3	
1	5	1	8	20
2	4	7	10	40
3	3	9	2	20
4	7	4	1	15
5	8	6	10	30
Budget	25	25	25	

Variables

$$x_j = \begin{cases} 1 & \text{si le projet } j \text{ est sélectionné,} \\ 0 & \text{sinon.} \end{cases}$$

Formulation

$$\begin{array}{rccccccc} \max\ z = & 20x_1 & +40x_2 & +20x_3 & +15x_4 & +30x_5 & & \\ \text{s.c.} & 5x_1 & +4x_2 & +3x_3 & +7x_4 & +8x_5 & \leq & 25 \\ & x_1 & +7x_2 & +9x_3 & +4x_4 & +6x_5 & \leq & 25 \\ & 8x_1 & +10x_2 & +2x_3 & +x_4 & +10x_5 & \leq & 25 \\ & x_1, & x_2, & x_3, & x_4, & x_5 & \in & \{0,1\} \end{array}$$

Exemple 17 (Problème avec coûts fixes).

– 3 compagnies de téléphone offrent des tarifs différents pour les communications longue distance.

Compagnie	Abonnement	Prix / minute
1	16	0.25
2	25	0.21
3	18	0.22

– Trouver le plan d'abonnement optimal pour 200 minutes de communication / mois.

Variables

x_i : minutes de communication avec la compagnie i.

$$y_i = \begin{cases} 1 & \text{si un abonnement est pris auprès de la compagnie } i, \\ 0 & \text{sinon.} \end{cases}$$

Formulation

$$
\begin{array}{rrrrrl}
\min\ z = & 0.25x_1 & +0.21x_2 & +0.22x_3 & +16y_1 + 25y_2 + 18y_3 & \\
\text{s.c.} & x_1 & +x_2 & +x_3 & = & 200 \\
& x_1 & & & \leq & 200y_1 \\
& & x_2 & & \leq & 200y_2 \\
& & & x_3 & \leq & 200y_3 \\
& x_1, & x_2, & x_3 & \geq & 0 \\
& y_1, & y_2, & y_3 & \in & \{0,1\}
\end{array}
$$

Exemple 18 (Voyageur de commerce).
- Un représentant doit visiter n villes une et une seule fois, et revenir à sa ville de départ, en minimisant le coût total du trajet.
- Le problème revient à trouver un tour de coût minimum passant une et une seule fois par chacun des noeuds d'un graphe. Le coût d'utilisation de l'arc (i,j) est c_{ij} .

Variables

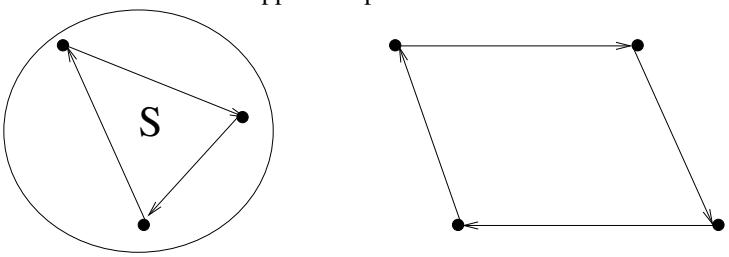
$$
x_{ij} = \begin{cases} 1 & \text{si l'arc } (i,j) \text{ appartient} \\ & \text{au tour optimal,} \\ 0 & \text{sinon.} \end{cases}
$$

Contraintes

$$
\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 1, \ldots, n
$$

$$
\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 1, \ldots, n
$$

Problème : apparition possible de *sous-tours*

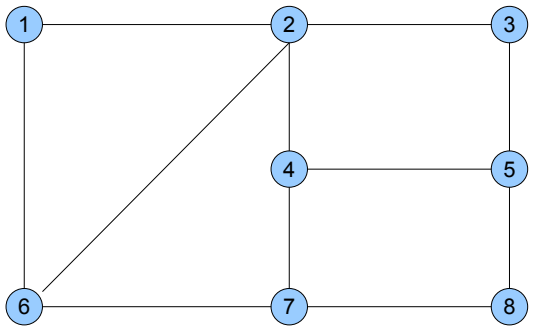


$$
\sum_{i,j \in S} x_{ij} \leq |S| - 1, \qquad \emptyset \neq S \neq \{1, \ldots, n\}.
$$

Formulation

$$
\begin{array}{rrll}
\min\ z = & \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} & & \\
\text{s.c.} & \displaystyle\sum_{j=1}^{n} x_{ij} = 1 & i = 1, \ldots, n & \\
& \displaystyle\sum_{i=1}^{n} x_{ij} = 1 & j = 1, \ldots, n & \\
& \displaystyle\sum_{i,j \in S} x_{ij} \leq |S| - 1 & \emptyset \neq S \neq \{1, \ldots, n\} & \\
& x_{ij} \in \{0,1\} & i = 1, \ldots, n,\ j = 1, \ldots, n &
\end{array}
$$

Exemple 19 (Problème de couverture). Le département de sécurité d'un campus veut installer des téléphones d'urgence. Chaque rue doit être servie par un téléphone, le but étant de minimiser le nombre de téléphones à installer (installation aux carrefours).



Formulation

$$
\begin{array}{rrrrrrrrrrrr}
\min\ z = & x_1 & +x_2 & +x_3 & +x_4 & +x_5 & +x_6 & +x_7 & +x_8 & & \\
\text{s.c.} & x_1 & +x_2 & & & & & & & \geq & 1 \\
& & x_2 & +x_3 & & & & & & \geq & 1 \\
& & & & x_4 & +x_5 & & & & \geq & 1 \\
& & & & & & x_6 & +x_7 & & \geq & 1 \\
& & & & & & & x_7 & +x_8 & \geq & 1 \\
& x_1 & & & & & +x_6 & & & \geq & 1 \\
& & x_2 & & & & +x_6 & & & \geq & 1 \\
& & x_2 & & +x_4 & & & & & \geq & 1 \\
& & & & x_4 & & & +x_7 & & \geq & 1 \\
& & & x_3 & & +x_5 & & & & \geq & 1 \\
& & & & & x_5 & & & +x_8 & \geq & 1 \\
& x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7, & x_8 & \in & \{0,1\}
\end{array}
$$

Contraintes disjonctives

– Dans un programme linéaire, toutes les contraintes doivent être satisfaites simultanément.
– Parfois, il est nécessaire de modéliser le fait qu'une contrainte parmi un ensemble doit être satisfaite. Si les contraintes de l'ensemble sont mutuellement incompatibles, on parle de contraintes disjonctives.

Exemple 20 (Contraintes disjonctives).
– Une machine est utilisée pour 3 tâches différentes. Pour chaque tâches i, une durée p_i et une date limite d_i sont données, ainsi qu'une pénalité par jour de retard.

Tâche	Durée	Limite	Pénalité
1	5	25	19
2	20	22	12
3	15	35	34

– Comment arranger les tâches sur la machine pour minimiser la pénalité totale ?
– Variables : x_i : temps de fin de la tâche i ($x_i \geq p_i$).
– Deux tâches i et j ne peuvent être éxécutées simultanément, donc

$$x_i \geq x_j + p_i \textit{ ou } x_j \geq x_i + p_j.$$

– Pour introduire ces contraintes disjonctives, nous faisons appel à des variables binaires auxiliaires :

$$y_{ij} = \begin{cases} 1 & \text{si } i \text{ précède } j \\ 0 & \text{si } j \text{ précède } i \end{cases}$$

$$x_i - x_j + My_{ij} \geq p_i$$
$$x_j - x_i + M(1 - y_{ij}) \geq p_j$$

– Contrainte de date limite : introduction d'une variable d'écart non restreinte $x_i + s_i = d_i$.
– Une pénalité apparaît uniquement si $s_i < 0$.
– Remplacement par deux variables non négatives représentant les parties positive et négative.

$$x_i + s_i^+ - s_i^- = d_i$$
$$s_i^+, s_i^- \geq 0$$

$$\text{Objectif :} \quad \min\ z = 19s_1^- + 12s_2^- + 34s_3^-$$

Formulation

$$\begin{array}{rrrrrrrrrrrrrrr}
\min\ z = & & & & & 19s_1^- & & +12s_2^- & & +34s_3^- & & \\
\text{s.c.} & x_1 & -x_2 & & & +My_{12} & & & & & \geq & 5 \\
& -x_1 & +x_2 & & & -My_{12} & & & & & \geq & 20 - M \\
& x_1 & & -x_3 & & & & +My_{13} & & & \geq & 5 \\
& -x_1 & & +x_3 & & & & -My_{13} & & & \geq & 15 - M \\
& & x_2 & -x_3 & & & & & & +My_{23} & \geq & 20 \\
& & -x_2 & +x_3 & & & & & & -My_{23} & \geq & 15 - M \\
& x_1 & & & & +s_1^+ - s_1^- & & & & & = & 25 \\
& & x_2 & & & & & +s_2^+ - s_2^- & & & = & 22 \\
& & & x_3 & & & & & & +s_3^+ - s_3^- & = & 35 \\
& x_1, & x_2, & x_3, & & s_1^+, s_1^-, & & s_2^+, s_2^-, & & s_3^+, s_3^- & \geq & 0 \\
& x_1 & & & & & & & & & \geq & 5 \\
& & x_2 & & & & & & & & \geq & 20 \\
& & & x_3 & & & & & & & \geq & 15 \\
& & & & & y_{12}, & & y_{13}, & & y_{23} & \in & \{0,1\}
\end{array}$$

7 Complexité des problèmes et efficacité des algorithmes

Problèmes faciles et difficiles

– La théorie de la complexité s'attache à classifier les problèmes selon leur difficulté.
– Un problème est *facile* s'il existe un algorithme efficace pour le résoudre.
– Exemples de problèmes "faciles" : programmation linéaire, affectation, plus courts chemins, ...
– Un problème est *difficile* s'il appartient à la classe des problèmes *NP-complets*, pour lesquels il est peu probable de trouver un jour un algorithme efficace.
– Exemple de problèmes "difficiles" : programmes en nombres entiers, voyageur de commerce, ...

Algorithmes efficaces et explosion combinatoire

– L'efficacité d'un algorithme est mesurée par l'ordre de grandeur du *nombre d'opérations élémentaires* qu'il effectue en fonction de la taille des données.
– Un algorithme sera *efficace* si le nombre d'opérations élémentaires est *polynomial* (i.e. borné supérieurement par un polynôme) en la taille de problème.

n	$\ln n + 1$	n	n^2	n^3	2^n
1	1.000	1	1	1	2
2	1.301	2	4	8	4
3	1.477	3	9	27	8
5	1.699	5	25	125	32
10	2.000	10	100	1000	1024
20	2.301	20	400	8000	1048576
50	2.699	50	2500	125000	1.12×10^{15}
100	3.000	100	10000	1000000	1.27×10^{30}

– Autre perspective : supposons que trois ordinateurs M_1, M_2 et M_3 effectuent respectivement 10000, 20000 et 40000 opérations par seconde. opérations par seconde.
– Quelle taille maximum de problème n peut on résoudre en une minute par des algorithmes effectuant respectivemet n, n^2, n^3 et 2^n opérations ?

Ordinateur	n	n^2	n^3	2^n
M_1	600000	775	84	19
M_2	1200000	1095	106	20
M_3	2400000	1549	134	21

8 Problèmes polynomiaux

8.1 Le problème d'affectation

Le problème d'affectation
– Exemple de problème combinatoire pour lequel il existe un algorithme efficace.
– La meilleure personne pour chaque tâche.
– n personnes doivent effectuer n tâches.
– Un coût c_{ij} est associé à l'affectation de la personne i à la tâche j.

Formulation du problème d'affectation

$$\begin{aligned} \min\ z = & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \text{s.c.} & \sum_{j=1}^{n} x_{ij} = 1 \quad i = 1,\dots,n \\ & \sum_{i=1}^{n} x_{ij} = 1 \quad j = 1,\dots,n \\ & x_{ij} \in \{0,1\} \quad i = 1,\dots,n,\ j = 1,\dots,n \end{aligned}$$

– Propriété : la relaxation linéaire du problème est toujours entière.
– Algorithme : *méthode hongroise*.

Exemple 21 (Problème d'affectation). Un père propose 3 travaux à ses enfants : tondre la pelouse, peindre le garage et laver la voiture. Il demande à chaque enfant combien il voudrait être payé pour chaque travail.

	Tondre	Peindre	Laver
John	15	10	9
Karen	9	15	10
Terri	10	12	8

Méthode hongroise
– Sélectionner le prix minimum dans chaque ligne.

	Tondre	Peindre	Laver	
John	15	10	9	9
Karen	9	15	10	9
Terri	10	12	8	8

– Soustraire ce prix de chaque ligne et sélectionner le prix minimum dans chaque colonne.

	Tondre	Peindre	Laver	
John	6	1	0	9
Karen	0	6	1	9
Terri	2	4	0	8
	0	1	0	

– Soustraire ce prix de chaque colonne.

	Tondre	Peindre	Laver	
John	6	0	0	9
Karen	0	5	1	9
Terri	2	3	0	8
	0	1	0	

- Les "0" en bleu donnent une affectation admissible optimale.
- L'optimalité est assurée par la valeur des variables duales (en rouge).
- Nombre d'opérations : $O(n^2)$.
- Il arrive que les valeurs nulles du tableau ne permettent pas de trouver de solution admissible.

1	4	6	3
9	7	10	9
4	5	11	7
8	7	8	5

0	3	2	2	1
2	0	0	2	7
0	1	4	3	4
3	2	0	0	5
0	0	3	0	

- Sélectionner un nombre minimum de lignes et colonnes couvrant tous les 0.

0	3	2	2	1
2	0	0	2	7
0	1	4	3	4
3	2	0	0	5
0	0	3	0	

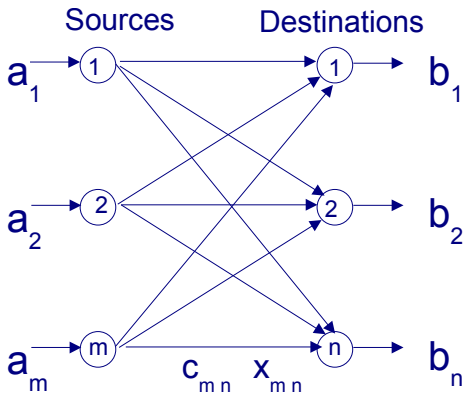
- Sélectionner le plus petit élément non couvert, le soustraire à tous les éléments non couverts et l'ajouter aux intersections.

0	2	1	1	2
3	0	0	2	7
0	0	3	2	5
4	2	0	0	5
-1	0	3	0	

- Répéter jusqu'à trouver une solution admissible.

8.2 Modèle de transport

- Un produit doit être transporté de *sources* (usines) vers des *destinations* (dépôts, clients).
- *Objectif :* déterminer la quantité envoyée de chaque source à chaque destination en minimisant les coûts de transport. Les coûts sont proportionnels aux quantités transportées.
- *Contraintes* d'offre limitée aux sources et de demande à satisfaire au destinations.



Exemple 22 (Modèle de transport).

– Une firme automobile a trois usines à Los Angeles, Detroit et New Orleans, et deux centres de distribution à Denver et Miami.
– Les capacités des trois usines sont de 1000, 1500 et 1200 respectivement, et les demandes aux centres de distribution sont de 2300 et 1400 voitures.
– Coûts :

	Denver	Miami
Los Angeles	80	215
Detroit	100	108
New Orleans	102	68

Formulation

$$
\begin{array}{rlllllll}
\min z = & 80x_{11} & +215x_{12} & +100x_{21} & +108x_{22} & +102x_{31} & +68x_{32} & \\
\text{s.c.} & & & & & & & \\
(Los\ Angeles) & x_{11} & +x_{12} & & & & & = 1000 \\
(Detroit) & & & x_{21} & +x_{22} & & & = 1500 \\
(New\ Orleans) & & & & & x_{31} & +x_{32} & = 1200 \\
(Denver) & x_{11} & & +x_{21} & & +x_{31} & & = 2300 \\
(Miami) & & x_{12} & & +x_{22} & & +x_{32} & = 1400 \\
& x_{11}, & x_{12}, & x_{21}, & x_{22}, & x_{31}, & x_{32} & \geq 0
\end{array}
$$

Représentation tableau

	Denver	Miami	Offre
Los Angeles	80 *1000*	215	1000
Detroit	100 *1300*	108 *200*	1500
New Orleans	102	68 *1200*	1200
Demande	2300	1400	

Problèmes non balancés
– Si l'offre n'est pas égale à la demande : modèle non balancé.
– Introduction d'une source ou destination artificielle.

	Denver	Miami	Offre
Los Angeles	80 *1000*	215	1000
Detroit	100 *1300*	108	1300
New Orleans	102	68 *1200*	1200
Artif.	0	0 *200*	200
Demande	2300	1400	

Variantes
Le modèle de transport n'est pas limité au transport de produits entre des sources et destinations géographiques.

Exemple 23 (Modèle de production).
– Une société fabrique des sacs à dos, pour lesquels la demande arrive de mars à juin et est de 100, 200, 180 et 300 unités, respectivement.
– La production pour ces mois est de 50, 180, 280 et 270, respectivement.
– La demande peut être satisfaite

1. par la production du mois courant ($40 / sac) ;

2. par la production d'un mois précédent (+ \$0.5 / sac / mois pour le stockage) ;
3. par la production d'un mois suivant (+ \$2 / sac / mois de pénalité de retard).

Correspondances avec le modèle de transport

Transport	Production - stocks
Source i	Période de production i
Destination j	Période de demande j
Offre à la source i	Capacité de production à la période i
Demande à la destination j	Demande pour la période j
Coût de transport de i à j	Coût unitaire (production + stock + pénalité) pour une production en période i pour la période j

Exemple 24 (Maintenance d'équipements).
– Une scierie prépare différents types de bois sur base d'un plan hebdomadaire.
– Pour satisfaire la demande, dépendant du type de bois, la scierie utilise un nombre donné de lames.
– Pour satisfaire la demande, deux possibilités :
  – acheter une lame (\$12) ;
  – faire aiguiser la lame (service express : \$6 en une nuit, sinon \$3 en 2 jours).
– Demande :

Lun	Mar	Mer	Jeu	Ven	Sam	Dim
24	12	14	20	18	14	22

Modèle de transport

8 sources : achat de nouvelles lames (offre = demande totale), 7 jours de la semaine (offre = nombre de lames utilisées).

8 destinations : 7 jours de la semaine (demande = nombre de lames utilisées) + surplus de lames non achetées / aiguisées (demande = nombre total de lames).

	Lun	Mar	Mer	Jeu	Ven	Sam	Dim	Surplus	Offre
Achat	12	12	12	12	12	12	12	0	124
Lun	10000	6	6	3	3	3	3	0	24
Mar	10000	10000	6	6	3	3	3	0	12
Mer	10000	10000	10000	6	6	3	3	0	14
Jeu	10000	10000	10000	10000	6	6	3	0	20
Ven	10000	10000	10000	10000	10000	6	6	0	18
Sam	10000	10000	10000	10000	10000	10000	6	0	14
Dim	10000	10000	10000	10000	10000	10000	10000	0	22
Demande	24	12	14	20	18	14	22	124	

Algorithme pour le problème de transport
Basé sur l'algorithme du simplexe en tenant compte de la structure du problème.

1. Détermination d'une solution de base admissible.
2. Détermination de la variable entrant en base.
3. Détermination de la variable sortant de base.

Exemple 25 (Algorithme pour le problème de transport).

	1	2	3	4	Offre
1	10	2	20	11	15
2	12	7	9	20	25
3	4	14	16	18	10
Demande	5	15	15	15	

Détermination d'une solution de base admissible
– Heuristiques "gloutonnes", pas besoin de méthode des deux phases.
– Variantes :

1. Coin Nord-Ouest
2. Méthode des moindres coûts
3. Approximation de Vogel (VAM)

Coin Nord-Ouest
Partir du coin supérieur gauche du tableau.

1. allouer le plus possible à la cellule courante et ajuster l'offre et la demande ;
2. se déplacer d'une cellule vers la droite (demande nulle) ou le bas (offre nulle) ;
3. répéter jusqu'au moment où toute l'offre est allouée.

Exemple 26 (Coin Nord-Ouest).

	1	2	3	4	Offre
1	10 *5*	2 *10*	20	11	15
2	12	7 *5*	9 *15*	20 *5*	25
3	4	14	16	18 *10*	10
Demande	5	15	15	15	

Coût : 520

Moindres coûts
Sélectionner la cellule de coût minimum.

1. allouer le plus possible à la cellule courante et ajuster l'offre et la demande ;
2. sélectionner la cellule de coût minimum ayant une demande et une offre non nulles ;
3. répéter jusqu'au moment où toute l'offre est allouée.

Exemple 27 (Moindres coûts).

	1	2	3	4	Offre
1	10	2 *15*	20	11	15
2	12	7	9 *15*	20 *10*	25
3	4 *5*	14	16	18 *5*	10
Demande	5	15	15	15	

Coût : 475

Approximation de Vogel (VAM)

1. pour chaque ligne (colonne) avec une offre (demande) non-nulle, calculer une pénalité égale à la différence entre les deux coûts les plus petits dans la ligne (colonne) ;
2. sélectionner la ligne ou colonne avec la pénalité maximale et sélectionner la cellule de coût minimum dans la ligne ou colonne ;
3. allouer le plus possible à la cellule courante ;
4. lorsqu'il ne reste qu'une ligne ou colonne : moindre coûts.

Exemple 28 (VAM).

	1	2	3	4	Offre
1	10	2	20	11	15
		15			
2	12	7	9	20	25
			15	*10*	
3	4	14	16	18	10
	5			*5*	
Demande	5	15	15	15	

Coût : 475

Formulation

$$\begin{aligned} \min\ z = & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} \\ \text{s.c.} \quad (u_i) & \sum_{j=1}^{n} x_{ij} = a_i \qquad i = 1,\dots,m \\ (v_j) & \sum_{i=1}^{m} x_{ij} = b_j \qquad j = 1,\dots,n \\ & x_{ij} \geq 0 \qquad i = 1,\dots,m,\ j = 1,\dots,n \end{aligned}$$

Problème dual

$$\begin{aligned} \max\ w = & \sum_{i=1}^{m} a_i u_i + \sum_{j=1}^{n} b_j v_j \\ \text{s.c.} \quad & u_i + v_j \leq c_{ij} \qquad i = 1,\dots,m,\ j = 1,\dots,n \end{aligned}$$

Adaptation du simplexe

Critère d'optimalité :

$$u_i + v_j - c_{ij} \leq 0$$

Complémentarité :

$$x_{ij} > 0 \Rightarrow u_i + v_j - c_{ij} = 0$$

Trois étapes :
1. détermination des variables duales (multiplicateurs) ;
2. vérification du critère d'optimalité et détermination de la variable entrante ;
3. détermination de la variable sortante pour préserver l'admissibilité et pivotage.

Détermination des variables duales

1. $m + n - 1$ équations à $m + n$ inconnues : fixer $u_1 = 0$.

2. Résoudre récursivement le système

$$u_i + v_j - c_{ij} = 0 \text{ pour tout } x_{ij} > 0.$$

Exemple 29 (Détermination des variables duales).

	1	2	3	4	Offre
1	10	2	20	11	15
	5	*10*			
2	12	7	9	20	25
		5	*15*	*5*	
3	4	14	16	18	10
				10	
Demande	5	15	15	15	

$$\begin{aligned} & & u_1 &= 0 \\ u_1 + v_1 = 10 & \Rightarrow & v_1 &= 10 \\ u_1 + v_2 = 2 & \Rightarrow & v_2 &= 2 \\ u_2 + v_2 = 7 & \Rightarrow & u_2 &= 5 \\ u_2 + v_3 = 9 & \Rightarrow & v_3 &= 4 \\ u_2 + v_4 = 20 & \Rightarrow & v_4 &= 15 \\ u_3 + v_4 = 18 & \Rightarrow & u_3 &= 3 \end{aligned}$$

Vérification du critère d'optimalité et détermination de la variable entrante

	1	**10**	2	**2**	3	**4**	4	**15**	Offre
1	10		2		20		11		15
0	*5*		*10*			-16		4	
2	12		7		9		20		25
5		3	*5*		*15*		*5*		
3	4		14		16		18		10
3		9		-9		-9	*10*		
Demande	5		15		15		15		

Détermination de la variable sortante pour préserver l'admissibilité et pivotage

Objectifs : 1. l'offre et la demande doivent continuer à être satisfaites ;

2. les quantités transportées doivent rester positives.

Méthode : 1. construction d'un *cycle* parcourant des variables en base en partant de et revenant à la variable entrante ;

2. déplacement le long de lignes et colonnes en alternant ajout et retrait d'une même quantité.

	1	**10**	2	**2**	3	**4**	4	**15**	Offre
1	10	$-\theta$	2	$+\theta$	20		11		15
0	*5*		*10*			-16		4	
2	12		7	$-\theta$	9		20	$+\theta$	25
5		3	*5*		*15*		*5*		
3	4	$+\theta$	14		16		18	$-\theta$	10
3		9		-9		-9	*10*		
Demande	5		15		15		15		

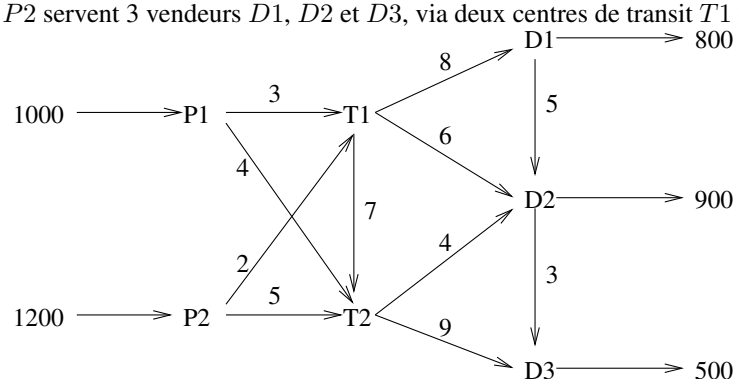
$$\theta = 5$$

	1	**1**	2	**2**	3	**4**	4	**15**	Offre
1	10		2	$-\theta$	20		11	$+\theta$	15
0		-9	*15*			-16		4	
2	12		7	$+\theta$	9		20	$-\theta$	25
5		-6	*0*		*15*		*10*		
3	4		14		16		18		10
3	*5*			-9		-9	*5*		
Demande	5		15		15		15		

$\theta = 10$

	1	**-3**	2	**2**	3	**4**	4	**11**	Offre
1	10		2		20		11		15
0		-13	*5*			-16	*10*		
2	12		7		9		20		25
5		-10	*10*		*15*			-4	
3	4		14		16		18		10
7	*5*			-5		-5	*5*		
Demande	5		15		15		15		

– Extension du modèle de transport : il est parfois nécessaire (ou moins cher) d'utiliser des *noeuds intermédiaires* pour le transport.
– Deux usines $P1$ et $P2$ servent 3 vendeurs $D1$, $D2$ et $D3$, via deux centres de transit $T1$ et $T2$.



Transformation en problème de transport

– 3 types de noeuds :

Noeuds d'offre purs : arcs sortants uniquement. offre = offre originale

Noeuds de demande purs : arcs entrants uniquement. demande = demande originale

Noeuds de transbordement : arcs entrants et sortants. offre/demande = offre/demande originale + *buffer*

– Les noeuds de transbordement sont à la fois sources et destinations pour le problème de transport.
– *Buffer :* quantité nécessaire pour transporter toute la demande à travers le noeud de transbordement.
– Dans notre exemple : $B = 2200$.

	T1	T2	D1	D2	D3	Offre
P1	3	4	M	M	M	1000
P2	2	5	M	M	M	1200
T1	0	7	8	6	M	2200
T2	M	0	M	4	9	2200
D1	M	M	0	5	M	2200
D2	M	M	M	0	3	2200
Demande	2200	2200	3000	3100	500	

9 Méthodes de Branch-and-Bound

9.1 Branch-and-Bound pour les problèmes en nombres entiers

- Les méthodes de branch-and-bound sont des méthodes basées sur une énumération "intelligente" des solutions admissibles d'un problème d'optimisation combinatoire.
- Idée : prouver l'optimalité d'une solution en partitionant l'espace des solutions.
- *"Diviser pour régner"*
- Application à la programmation linéaire en nombres entiers : utilise toute la puissance de la programmation linéaire pour déterminer de bonnes bornes.
- On appelle *relaxation linéaire* d'un programme linéaire en nombres entiers le programme linéaire obtenu en supprimant les contraintes d'intégralité sur les variables.

Programme en nombres entiers

$$\begin{array}{lll} (P) & \max & c^T x \\ & \text{s.c.} & Ax \leq b \\ & & x \geq 0, \text{ entier.} \end{array}$$

Relaxation linéaire

$$\begin{array}{lll} (LP) & \max & c^T x \\ & \text{s.c.} & Ax \leq b \\ & & x \geq 0. \end{array}$$

Propriétés de la relaxation linéaire

- La valeur de la solution optimale de LP est une *borne supérieure* sur la valeur de la solution optimale de P.
- La valeur d'une solution admissible de P fournit une *borne inférieure* sur la valeur de la solution optimale de P.
- Si la solution optimale de LP est entière (donc admissible pour P), elle est également la solution optimale de P.

Branchement

- Si la solution de LP n'est pas entière, soit x_i une variable prenant une valeur fractionnaire x_i^* dans la solution optimale de LP.
- Le problème peut être divisé en deux sous-problèmes en imposant

$$x_i \leq \lfloor x_i^* \rfloor \qquad \text{ou} \qquad x_i \geq \lfloor x_i^* \rfloor + 1$$

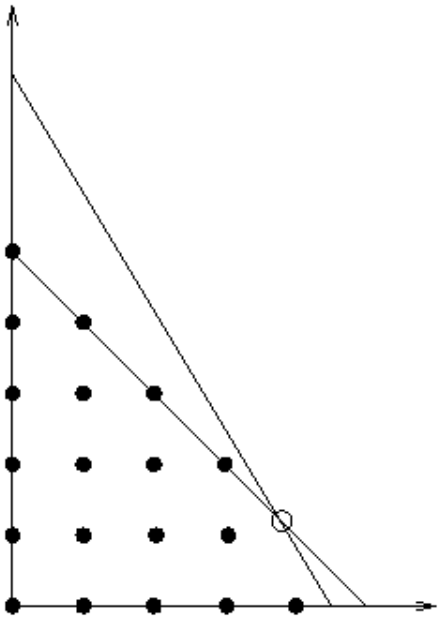
  où $\lfloor x_i^* \rfloor$ est le plus grand entier inférieur à x_i^*.
- La solution optimale de P est la meilleure des solutions optimales des deux problèmes

$$\begin{array}{lll} (P_1) & \max & c^T x \\ & \text{s.c.} & Ax \leq b \\ & & x_i \leq \lfloor x_i^* \rfloor \\ & & x \geq 0, \text{ entier.} \end{array} \qquad \begin{array}{lll} (P_2) & \max & c^T x \\ & \text{s.c.} & Ax \leq b \\ & & x_i \geq \lfloor x_i^* \rfloor + 1 \\ & & x \geq 0, \text{ entier.} \end{array}$$

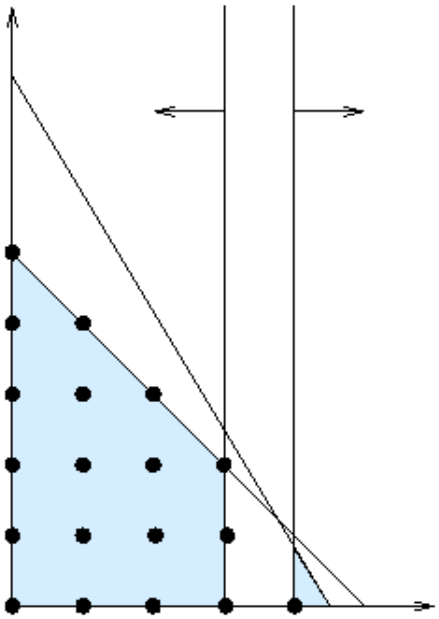
Exemple 30 (Branch-and-Bound).

$$\begin{array}{rrrrl} \max z = & 5x_1 & +4x_2 & & \\ \text{s.c.} & x_1 & +x_2 & \leq 5 & \\ & 10x_1 & +6x_2 & \leq 45 & \\ & x_1, & x_2 & \geq 0, \text{ entiers.} & \end{array}$$

$$\begin{array}{ll} (LP) & x_1 = 3.75,\ x_2 = 1.25 \\ & z = 23.75 \end{array}$$



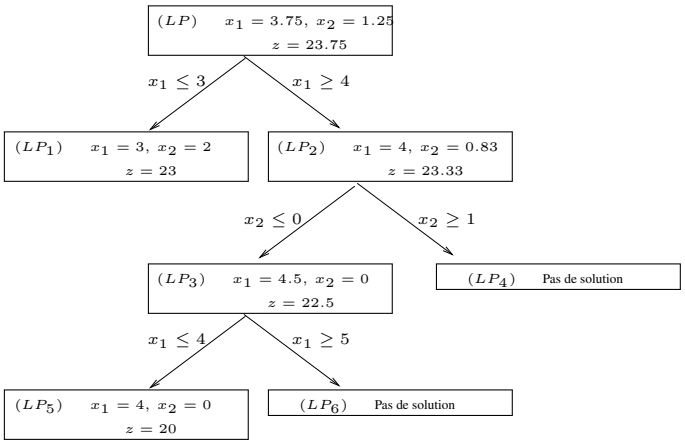
– La solution de LP_1 est une solution admissible de P et donc $z = 23$ est une borne inférieure sur la valeur de la solution optimale de P.
– Le noeud correspondant peut être éliminé vu qu'une solution entière optimale satisfaisant $x_1 \leq 3$ a été trouvée (solution de P_1).



– La valeur de la solution de LP, $z = 23.75$ est une borne supérieure sur la valeur de la solution optimale de P.

– Vu que tout les coefficients sont entiers, on peut en déduire que la valeur de la solution optimale de P est inférieure ou égale à 23.
– La solution de P_1 est donc optimale pour P.

Règles de branchement
– Il n'y a pas de règle générale pour le choix de la variable de branchement et de la branche à examiner en premier.
– Ce choix peut avoir un impact important sur le nombre de noeuds à examiner dans l'arbre de branch-and-bound.
– Exemple : branchement d'abord du côté $\geq$.



9.2 Branch-and-bound pour le voyageur de commerce

Formulation (rappel)

$$
\begin{aligned}
\min\ z = \quad & \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} & \\
\text{s.c.} \quad & \sum_{j=1}^{n} x_{ij} = 1 & i = 1,\dots,n \\
& \sum_{i=1}^{n} x_{ij} = 1 & j = 1,\dots,n \\
& \sum_{i,j\in S} x_{ij} \leq |S| - 1 & \emptyset \neq S \neq \{1,\dots,n\} \\
& x_{ij} \in \{0,1\} & i = 1,\dots,n,\ j = 1,\dots,n
\end{aligned}
$$

– Si on retire les contraintes d'élimination de sous-tours, on obtient le problème d'affectation.
– Cette relaxation a une solution entière qui peut être obtenue par exemple avec la méthode hongroise.
– Le branchement est effectué de manière à éliminer les sous-tours.
– La valeur de la solution optimale du problème d'affectation (AP) est une *borne inférieure* sur la valeur de la solution optimale du TSP.
– Le coût d'un tour fournit une *borne supérieure* sur la valeur de la solution optimale.
– Si la solution optimale de AP est un tour (i.e. sans sous-tour), elle est également la solution optimale du TSP.
– Si un sous-tour apparaît :

$$x_{i_1 i_2} = x_{i_2 i_3} = x_{i_3 i_4} = \cdots = x_{i_{k-1} i_k} = x_{i_k i_1} = 1.$$

– Dans une solution admissible, un de ces arcs doit être absent, donc

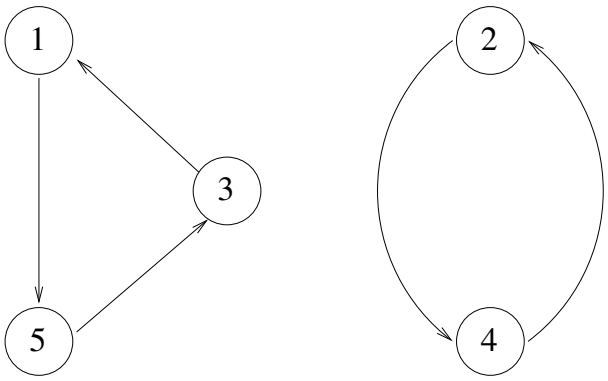
$$x_{i_1 i_2} = 0 \text{ ou } x_{i_2 i_3} = 0 \text{ ou } \ldots \text{ ou } x_{i_{k-1} i_k} = 0 \text{ ou } x_{i_k i_1} = 0.$$

– Chacune de ces conditions va correspondre à une branche de l'arbre de branch-and-bound.

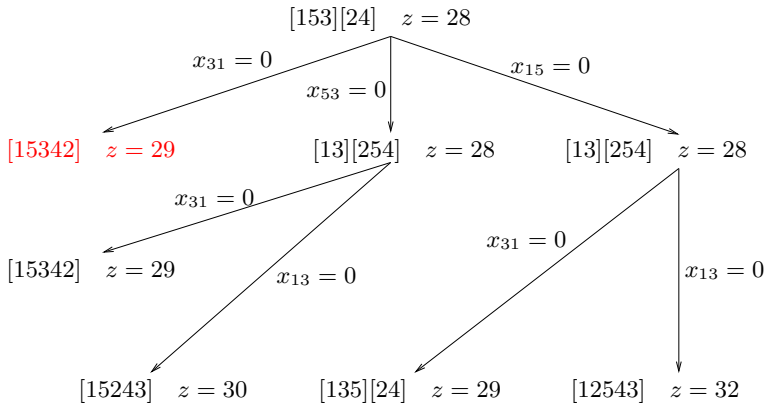
Exemple 31 (Voyageur de commerce).

	1	2	3	4	5
1	∞	8	2	10	3
2	9	∞	20	9	7
3	6	40	∞	7	5
4	8	4	2	∞	5
5	9	10	6	9	∞

Solution du problème d'affectation



$[153][24] \quad z = 28$



9.3 Branch-and-bound pour les contraintes disjonctives

– On peut utiliser une approche classique pour les problèmes en nombres entiers en introduisant des variables supplémentaires (voir formulation en début de partie).
– On peut également prendre comme relaxation le problème sans les contraintes disjonctives, et brancher sur les contraintes non satisfaites.

Exemple 32 (Contraintes disjonctives).
– Une machine est utilisée pour 3 tâches différentes. Pour chaque tâches i, une durée p_i et une date limite d_i sont données, ainsi qu'une pénalité par jour de retard.

Tâche	Durée	Limite	Pénalité
1	5	25	19
2	20	22	12
3	15	35	34

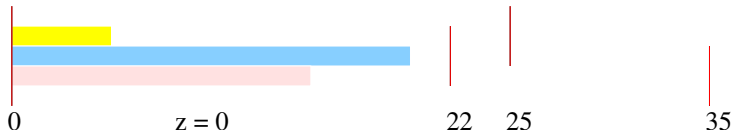
– Comment arranger les tâches sur la machine pour minimiser la pénalité totale ?

Relaxation du problème

$$
\begin{array}{rlllllll}
\min\ z = & & 19s_1^- & +12s_2^- & +34s_3^- & & \\
\text{s.c.} & x_1 + s_1^+ - s_1^- & & & & = & 25 \\
& x_2 & & +s_2^+ - s_2^- & & = & 22 \\
& x_3 & & & +s_3^+ - s_3^- & = & 35 \\
& x_1, x_2, x_3, \ s_1^+, \ s_1^-, & & s_2^+, \ s_2^-, & s_3^+, \ s_3^- & \geq & 0 \\
& x_1 & & & & \geq & 5 \\
& x_2 & & & & \geq & 20 \\
& x_3 & & & & \geq & 15
\end{array}
$$

– Branchement sur $x_i \geq x_j + p_i$ ou $x_j \geq x_i + p_j$.
– Solution de la relaxation : ordonnancement “au plus tôt”.

Noeud 0



Relaxation : 0 Borne sup. : $+\infty$.

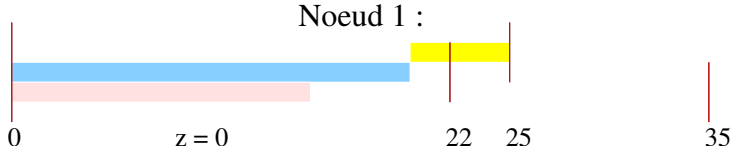
Noeuds à examiner :

$$
\begin{array}{ll}
1: & x_1 - x_2 \geq 5 \\
2: & x_2 - x_1 \geq 20
\end{array}
$$

Noeud 1

$$1: \quad x_1 - x_2 \geq 5$$

Noeud 1 :



Relaxation : 0 Borne sup. : $+\infty$.

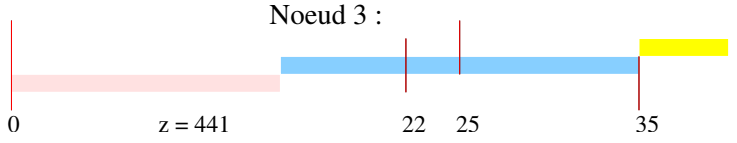
Noeuds à examiner :

$$
\begin{array}{ll}
2: & x_2 - x_1 \geq 20 \\
3: & x_1 - x_2 \geq 5 \\
 & x_2 - x_3 \geq 20 \\
4: & x_1 - x_2 \geq 5 \\
 & x_3 - x_2 \geq 15
\end{array}
$$

Noeud 3

$$
\begin{array}{ll}
3: & x_1 - x_2 \geq 5 \\
 & x_2 - x_3 \geq 20
\end{array}
$$

Noeud 3 :



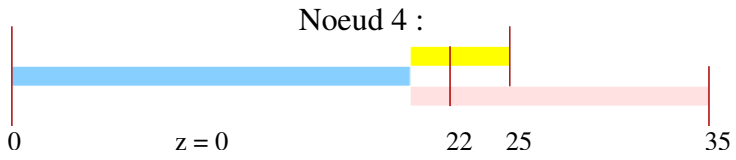
Relaxation : 441 Borne sup. : 441

Noeuds à examiner :

$$\begin{array}{ll} 2: & x_2 - x_1 \geq 20 \\ 4: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \end{array}$$

Noeud 4

$$\begin{array}{ll} 4: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \end{array}$$



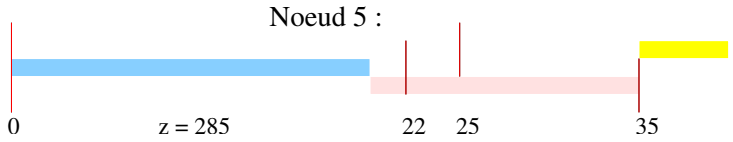
Relaxation : 0 Borne sup. : 441

Noeuds à examiner :

$$\begin{array}{ll} 2: & x_2 - x_1 \geq 20 \\ 5: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \\ & x_1 - x_3 \geq 5 \\ 6: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \\ & x_3 - x_1 \geq 15 \end{array}$$

Noeud 5

$$\begin{array}{ll} 5: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \\ & x_1 - x_3 \geq 5 \end{array}$$



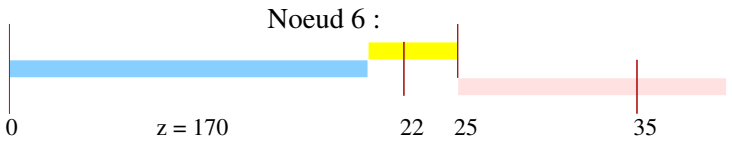
Relaxation : 285 Borne sup. : 285

Noeuds à examiner :

$$\begin{array}{ll} 2: & x_2 - x_1 \geq 20 \\ 6: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \\ & x_3 - x_1 \geq 15 \end{array}$$

Noeud 6

$$\begin{array}{ll} 6: & x_1 - x_2 \geq 5 \\ & x_3 - x_2 \geq 15 \\ & x_3 - x_1 \geq 15 \end{array}$$



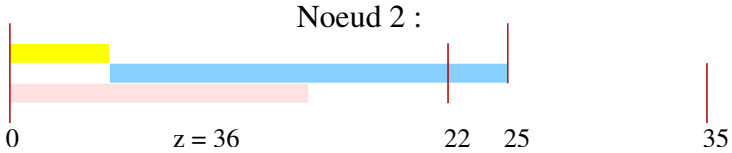
Relaxation : 170 Borne sup. : 170

Noeuds à examiner :

$$2 : \quad x_2 - x_1 \geq 20$$

Noeud 2

$$2 : \quad x_2 - x_1 \geq 20$$



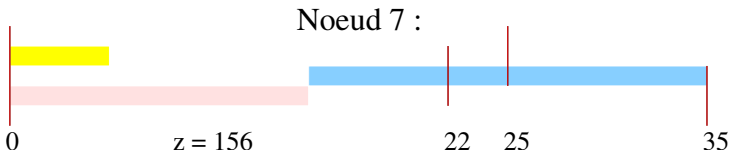
Relaxation : 36 Borne sup. : 170

Noeuds à examiner :

$$\begin{aligned} 7 : \quad & x_2 - x_1 \geq 20 \\ & x_2 - x_3 \geq 20 \\ 8 : \quad & x_2 - x_1 \geq 20 \\ & x_3 - x_2 \geq 15 \end{aligned}$$

Noeud 7

$$\begin{aligned} 7 : \quad & x_2 - x_1 \geq 20 \\ & x_2 - x_3 \geq 20 \end{aligned}$$



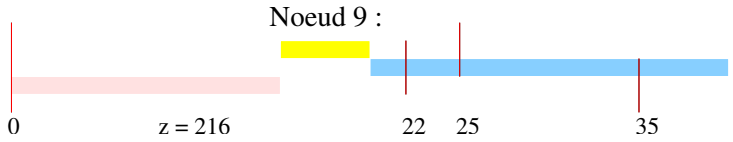
Relaxation : 156 Borne sup. : 170

Noeuds à examiner :

$$\begin{aligned} 8 : \quad & x_2 - x_1 \geq 20 \\ & x_3 - x_2 \geq 15 \\ 9 : \quad & x_2 - x_1 \geq 20 \\ & x_2 - x_3 \geq 20 \\ & x_1 - x_3 \geq 5 \\ 10 : \quad & x_2 - x_1 \geq 20 \\ & x_2 - x_3 \geq 20 \\ & x_3 - x_1 \geq 15 \end{aligned}$$

Noeud 9

$$
\begin{aligned}
9: \quad & x_2 - x_1 \geq 20 \\
& x_2 - x_3 \geq 20 \\
& x_1 - x_3 \geq 5
\end{aligned}
$$



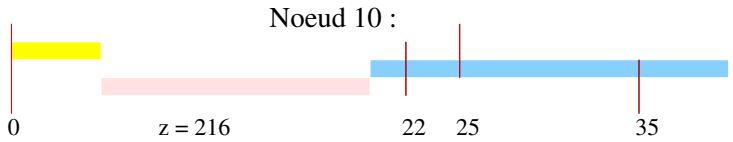
Relaxation : 216 Borne sup. : 170

Noeuds à examiner :

$$
\begin{aligned}
8: \quad & x_2 - x_1 \geq 20 \\
& x_3 - x_2 \geq 15 \\
10: \quad & x_2 - x_1 \geq 20 \\
& x_2 - x_3 \geq 20 \\
& x_3 - x_1 \geq 15
\end{aligned}
$$

Noeud 10

$$
\begin{aligned}
10: \quad & x_2 - x_1 \geq 20 \\
& x_2 - x_3 \geq 20 \\
& x_3 - x_1 \geq 15
\end{aligned}
$$



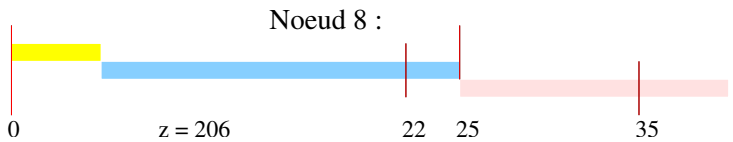
Relaxation : 216 Borne sup. : 170

Noeuds à examiner :

$$
\begin{aligned}
8: \quad & x_2 - x_1 \geq 20 \\
& x_3 - x_2 \geq 15
\end{aligned}
$$

Noeud 8

$$
\begin{aligned}
8: \quad & x_2 - x_1 \geq 20 \\
& x_3 - x_2 \geq 15
\end{aligned}
$$



Relaxation : 206 Borne sup. : 170

Solution optimale : 170 Ordre des tâches : 2, 1, 3 (noeud 6)

10 Méthodes heuristiques

10.1 Introduction

- La plupart des problèmes pratiques sont NP-complets.
- Nécessité de trouver des "bonnes" solutions rapidement.
- Heuristiques ou algorithmes d'approximation.

Raisons de choisir une heuristique
- Une solution doit être trouvée rapidement (secondes / minutes).
- Instance trop grande ou compliquée : impossible à formuler comme un problème en nombre entiers de taille raisonnable.
- Impossibilité pour le Branch-and-Bound de trouver une (bonne) solution admissible.
- Pour certaines classes de problèmes : trouver des solutions admissibles est facile (structure du problème) mais une approche généraliste de programmation en nombres entiers n'est pas efficace.

Questions à se poser...
- Doit-on accepter n'importe quelle solution, ou doit-on se demander a posteriori à quelle distance de l'optimalité on se trouve ?
- Peut-on garantir a priori que l'heuristique va produire une solution à ϵ (ou $\alpha\%$) de l'optimal ?
- Peut-on dire a priori que, pour la classe de problèmes considérée, l'heuristique va produire en moyenne une solution à $\alpha\%$ de l'optimal ?

Contexte général
Problème d'optimisation combinatoire

$$\begin{array}{ll} \min & F(x) \\ \text{s.c.} & x \in X. \end{array}$$

avec F une fonction à valeur réelles définie sur X,
X l'ensemble des solutions admissibles.

Hypothèse
X de trop grande taille pour permettre l'énumération.

10.2 Heuristiques de construction

- Objectif : construire une (bonne) solution admissible.
- Se basent sur la structure du problème pour générer une solution.
- Généralement : méthodes gloutonnes. Construisent la solution en ajoutant élément par élément, choix définitifs (pas de retour en arrière).
- Défaut : méthodes "aveugles", un mauvais choix fait en cours de construction ne peut pas être "défait".
- Exemples :
  - problème de transport (coin nord-ouest, moindre coûts, VAM)
  - voyageur de commerce (plus proche voisin, meilleure insertion)
  - heuristique gloutonne pour le problème de sac-à-dos.

Voyageur de commerce : plus proche voisin
- partir d'un sommet quelconque, par exemple le sommet 1
- tant qu'il reste des sommets libres faire :
  - connecter le dernier sommet atteint au sommet libre le plus proche
- relier le dernier sommet au sommet 1

Exemple 33 (Voyageur de commerce).

	1	2	3	4	5
1	∞	8	2	10	3
2	9	∞	20	9	7
3	6	40	∞	7	5
4	8	4	2	∞	5
5	9	10	6	9	∞

Tour : $1 \to 3 \to 5 \to 4 \to 2 \to 1$
Coût : 29
Remarque : si démarrage du noeud 2 :
Tour : $2 \to 5 \to 3 \to 1 \to 4 \to 2$
Coût : 33

Voyageur de commerce : meilleure insertion
– partir d'un cycle μ réduit à une boucle sur le sommet 1 (par exemple)
– tant qu'il y a des sommets libres faire :

– pour tous les sommets libres j, chercher la position d'insertion entre 2 sommets i, k du cycle μ minimisant $M = c_{ij} + c_{jk} - c_{ik}$
– insérer le sommet qui minimise l'accroissement du coût

Exemple 34 (Voyageur de commerce).

	1	2	3	4	5
1	∞	8	2	10	3
2	9	∞	20	9	7
3	6	40	∞	7	5
4	8	4	2	∞	5
5	9	10	6	9	∞

1. $1 \to 1$, $k = 3$, $M = 8$
2. $1 \to 3 \to 1$, $k = 5$, $M = 7$
3. $1 \to 5 \to 3 \to 1$, $k = 4$, $M = 5$
4. $1 \to 5 \to 4 \to 3 \to 1$, $k = 2$, $M = 10$
5. $1 \to 5 \to 2 \to 4 \to 3 \to 1$, coût : 30.

Le problème de sac-à-dos
– Le problème de sac-à-dos revient à décider quels objets mettre dans un contenant ayant une capacité donnée W de manière à maximiser le profit total des objets choisis.
– Chaque objet a un profit $p_i \geq 0$ et un poids $w_i \geq 0$.

Problème de sac-à-dos

$$\begin{aligned} z = \max \quad & \sum_{j=1}^{n} p_j x_j \\ \text{s.c.} \quad & \sum_{j=1}^{n} w_j x_j \leq W \\ & x_j \in \{0,1\} \qquad j = 1, \ldots, n \end{aligned}$$

Heuristique gloutonne pour le problème de sac-à-dos
Idée de base : Les objets les plus intéressants sont ceux qui ont un petit poids pour un grand profit, i.e. un rapport $\frac{p_i}{w_i}$ grand.

Heuristique gloutonne pour le problème de sac-à-dos

1. Ordonner les objets par ordre décroissant de $\frac{p_i}{w_i} \to$ ordre $i_1, \ldots, i_n$.
2. $S = \emptyset$.

3. Pour $t = 1, \ldots, n$ faire :
   Si $w(S) + w_{i_t} \leq W$, alors $S := S \cup \{i_t\}$. [1]

Analyse de l'heuristique

Résolution de la relaxation linéaire

1. Ordonner les objets par ordre décroissant de $\frac{p_i}{w_i} \rightarrow$ ordre $i_1, \ldots, i_n$.
2. $S = \emptyset$, $t = 1$.
3. Tant que $w(S) + w_{i_t} \leq W$:
   $S := S \cup \{i_t\}$.
4. Compléter la capacité avec la fraction nécessaire de l'objet i_t.

– Soit S la solution de l'heuristique gloutonne, z_{LP} la valeur de la relaxation linéaire et $p_0 = \max_{i=1,\ldots,n} p_i$.
– Considérons $\overline{z} = \max\{p(S), p_0\}$.
–

$$\overline{z} = \max\{p(S), p_0\} \geq \frac{p(S) + p_0}{2} \geq \frac{z_{LP}}{2} \geq \frac{z_{OPT}}{2}$$

10.3 Recherche locale

– Heuristiques de construction : utiles pour trouver une première solution admissible, mais souvent de mauvaise qualité.
– Idée : essayer d'améliorer cette solution en tenant compte de la structure du problème.
– Etant donné une solution, définition d'une notion de voisinage de cette solution.
– Méthode de descente : recherche du meilleur voisin jusqu'au moment où aucune amélioration n'est possible.
– Pour toute solution $x \in X$, on définit $V(x) \subseteq X$ comme étant l'ensemble des solutions voisines de x ($x \notin V(x)$).

Algorithme de descente

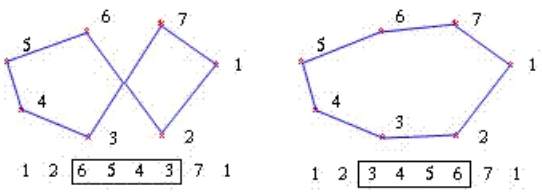
Initialisation : $x_0 \in X$ solution initiale.

Etape n **:** soit $x_n \in X$ la solution courante ;
sélectionner $x^* \in V(x_n)$;
si $F(x^*) \leq F(x_n)$:
$x_{n+1} := x^*$; passer à l'étape $n + 1$.
sinon x_n meilleure solution trouvée ; stop.

En général (si pas trop long), choix du meilleur élément du voisinage.

Voyageur de commerce : k-opt
Voisins d'un tour : tous les tours qui peuvent s'obtenir en retirant k arêtes et en reconnectant de la meilleurs manière possible.

Exemple 35 (2-opt).



1. Notation : $w(S) := \sum_{i \in S} w_i$

10.4 Méta-heuristiques

– Inconvénient principal des algorithmes de descente : s'arrête, le plus souvent, dans un optimum local et non dans un optimum global.
– Idée générale des méta-heuristiques : autoriser une détérioration temporaire de l'objectif, permettant ainsi de quitter des minimums locaux tout en maintenant en général une "pression" favorisant les solutions qui améliorent l' objectif.

Caractéristiques des méta-heuristiques
– plus simples et plus rapides à mettre en oeuvre quand l'exigence de qualité de la solution n'est pas trop importante ;
– plus souples dans les problèmes réels (contraintes non formulées dès le départ) ;
– s'adaptent plus facilement à des contraintes additionnelles venant du terrain ;
– fournissent des solutions et des bornes qui peuvent être utiles dans des méthodes exactes.

Le recuit simulé (simulated annealing)
Idée de base provient de l'opération de recuit (annealing), courante en sidérurgie et dans l'industrie du verre.

Recuit en sidérurgie
Après avoir fait subir des déformations au métal, on réchauffe celui-ci à une certaine température, de manière à faire disparaître les tensions internes causées par les déformations, puis on laisse refroidir lentement.
L'énergie fournie par le réchauffement permet aux atomes de se déplacer légèrement et le refroidissement lent fige peu à peu le système dans une structure d'énergie minimale.

Application du recuit simulé à l'optimisation
– Modification de l'heuristique de descente :
– au lieu de ne permettre que des mouvements (des changements de solution courante) qui diminuent l'énergie (la fonction objectif), on autorise des augmentations, même importantes, de l'énergie au début de l'exécution de l'algorithme ;
– puis, à mesure que le temps passe, on autorise ces augmentations de plus en plus rarement (la température baisse).
– Finalement, le système "gèle" dans un état d'énergie minimale.

Recuit simulé

Initialisation : $x_0 \in X$ solution initiale, $\hat{F} := F(x_0)$.

Etape n **:** soit $x_n \in X$ la solution courante ;
tirer au sort une solution $x^* \in V(x_n)$;
si $F(x^*) \leq F(x_n)$: $x_{n+1} := x^*$;
si $F(x^*) < \hat{F}$: $\hat{F} := F(x^*)$
sinon, tirer un nombre q au hasard entre O et 1 ;
si $q \leq p$: $x_{n+1} := x^*$;
sinon : $x_{n+1} := x_n$;
si la règle d'arrêt n'est pas satisfaite,
passer à l'étape $n + 1$;
sinon, stop.

Paramètres du recuit simulé
– La probabilité p est généralement une fonction $p(T, \Delta F)$ dépendant d'un paramètre T (température), et de la dégradation de l'objectif $\Delta F = F(x^*) - F(x_n)$ résultant du remplacement de x_n par x^* comme solution courante.
– Analogie avec les systèmes physiques (distribution de Boltzmann) :

$$p(T, \Delta F) = e^{-\frac{\Delta F}{T}}$$

– Evolution de la température : diminue toutes les L itérations.

$$T_k = \alpha T_{k-1} = \alpha^k T_0.$$

– T_0 choisi par simulation de sorte qu'au début de l'exécution de l'algorithme, des solutions moins bonnes que la solution courante soient aisément acceptées. En général, on fixe un taux d'acceptation initial moyen des solutions plus mauvaises que la solution courante.
– Exemple : taux fixé à 50%, évaluer par simulation la détérioration moyenne $< \Delta F >$ de F (en partant d'une solution initiale fixe).

$$T_O = \frac{< \Delta F >}{\ln 2} (\Rightarrow p = 0.5)$$

– La longueur L du palier de température doit être déterminée en tenant compte de la taille des voisinages ; elle devrait augmenter avec la taille du problème.
– En pratique, la détermination de L est expérimentale.
– Le paramètre de décroissance géométrique de la température α est fixé, le plus souvent, aux alentours de 0.9 ou 0.95.
– Critère d'arrêt : nombre maximal d'itérations ou système est "gelé" : e.g. si la fonction F a diminué de moins de 0.1 % pendant cent paliers consécutifs.

Quelques observations sur le recuit simulé
– Voyageur de commerce : pas concurrentiel par rapport à d'autres approches.
– En général : améliore les solutions d'une descente pure, mais très coûteux en temps calcul.
– Résultat théorique : sous certaines conditions, converge avec probabilité 1 vers une solution optimale.
– Résultat asymptotique (nombre d'itérations tendant à l'infini), peu de pertinence en pratique.

Recherche tabou
– Emprunte certains de ses concepts à l'intelligence artificielle (notion et utilisation de mémoire).
– Variante de la recherche locale.
– Partant de la solution courante x_n à l'étape n, on calcule la meilleure solution x^* dans un sous-voisinage V^* de $V(x_n)$. Cette solution devient la nouvelle solution courante x_{n+1}, qu'elle soit meilleure ou moins bonne que la précédente.
– Caractère non monotone pour éviter de rester coincé dans des minimums locaux.
– MAIS nécessité d'un mécanisme qui évite le cyclage.

Listes tabou
Pour éviter le cyclage, on définit une ou plusieurs listes, les listes tabou, qui gardent en mémoire les dernières solutions rencontrées ou des caractéristiques de celles-ci.
Le sous-voisinage V^* de $V(x_n)$ exclut dès lors les solutions rencontrées récemment ou les solutions ayant des caractéristiques communes avec celles-ci.

Exemple 36 (Voyageur de commerce).
– Villes : $\{A, B, C, D, E\}$
– Voisinage : 2-échange (échange de deux villes dans l'ordre des visites)
– Liste tabou : positions échangées dans le cycle.
– Solutions visitées :

$$\begin{aligned} x_n &= (A, B, C, D, E) \\ x_{n+1} &= (A, D, C, B, E) \qquad ((2,4) \text{ devient tabou}) \\ x_{n+2} &= (B, D, C, A, E) \qquad ((1,4) \text{ devient tabou}) \\ x_{n+3} &= (B, C, D, A, E) \qquad ((2,3) \text{ devient tabou}) \\ x_{n+4} &= (B, C, D, E, A) \qquad ((4,5) \text{ devient tabou}) \end{aligned}$$

– $x_{n+4} \equiv x_n$ mais OK si x_{n+5} peut être $\neq x_{n+1}$.
– Si la liste est de longueur 4, les solutions

$$(B, E, D, C, A), (E, C, D, B, A), (B, D, C, E, A), (B, C, D, A, E)$$

sont exclues pour x_{n+5}.

Critères d'aspiration
Puisque les listes tabou écartent des solutions non rencontrées, on peut envisager de négliger le statut tabou de certaines solutions si un avantage suffisant en résulte.
Ceci est implémenté à l'aide de critères d'aspiration. Par exemple, on acceptera pour x_{n+1} une solution x^* tabou si x^* donne à la fonction objectif F une valeur meilleure que toutes celles obtenues jusqu'à présent.

Recherche tabou

Initialisation : $x_0 \in X$ solution initiale, $\hat{F} := F(x_0)$, k = longueur de la liste tabou..

Etape n **:** soit $x_n \in X$ la solution courante ;
sélectionner, dans un sous-voisinage V^* de $V(x_n)$, la meilleure solution $x^* \in$ qui soit :
non tabou ou
tabou, mais satisfaisant un critère d'aspiration ;
$x_{n+1} := x^*$;
si $F(x^*) < \hat{F} : \hat{F} := F(x^*)$
mettre à jour la liste tabou ;
si la règle d'arrêt n'est pas satisfaite,
passer à l'étape $n + 1$;
sinon, stop.

Stratégies avancées de la recherche tabou
- Succession de phases d'intensification et de phases de diversification de la recherche.
- Pénalités ou bonifications introduites dans la fonction objectif pour favoriser ou défavoriser certains types de solutions.
- Adaptation de l'ensemble des solutions candidates V^*.
- Oscillation stratégique.

10.5 Algorithmes génétiques

- Conçus par Holland (1975) comme un modèle de système adaptatif complexe capable de simuler, notamment, l'évolution des espèces.
- Presque immédiatement après, appliqués à l'optimisation de fonctions de variables réelles. Par la suite, de très nombreux problèmes d'optimisation combinatoire ont été traités.
- Se distinguent du recuit et de la recherche tabou par le fait qu'ils traitent et font évoluer une population de solutions.
- Au cours d'une itération, les solutions de la population courante interagissent pour fournir la génération suivante (métaphore de la reproduction sexuée).

Description d'un algorithme génétique de base
- Les solutions sont codées de manière appropriée. Un codage élémentaire pour un problème de programmation mathématique en variables binaires est un vecteur x de 0 et de 1, où chaque composante x_j, $j = 1, \ldots, N$ représente la valeur prise par une variable.
- Pour le problème du voyageur de commerce, un codage plus usuel sera une liste ordonnée des noms (ou labels) des N villes.
- Un vecteur codant une solution est souvent appelé chromosome et ses coordonnées ou sites sont appelés gènes.
- Le choix d'un codage approprié est très important pour l'efficacité des opérateurs qui seront appliqués pour faire évoluer les solutions.
- Population initiale de solutions $X^{(0)}$ (taille constante au cours de l'évolution).
- Fonction d'évaluation des solutions : en général, croissante avec la qualité de la solution (fitness function, mesurant la "santé" de l'individu solution).
- Dans un problème de maximisation (respectivement, de minimisation), ce peut être la fonction objectif (respectivement, l'opposé de la fonction objectif).
- Pour des raisons d'efficacité de l'algorithme, on peut être amené à choisir la fonction d'évaluation de manière plus sophistiquée, mais elle sera toujours croissante (respectivement, décroissante) en la valeur de l'objectif dans un problème de maximisation (respectivement, de minimisation).

Algorithme génétique

Initialisation : $X^{(0)} \subset X$, population initiale.

Etape n **:** $X^{(n)} \subset X$, population courante ;

- sélectionner dans $X^{(n)}$ un ensemble de paires de solutions de haute qualité ;
- appliquer à chacune des paires de solutions sélectionnées un opérateur de croisement qui produit une ou plusieurs solutions enfants ;
- remplacer une partie de $X^{(n)}$ formée de solutions de basse qualité par des solutions "enfants" de haute qualité ;
- appliquer un opérateur de mutation aux solutions ainsi obtenues ; les solutions éventuellement mutées constituent la population $X^{(n+1)}$;

si la règle d'arrêt n'est pas satisfaite,
passer à l'étape $n+1$;
sinon, stop.

Sélection

- La sélection - aussi bien celle des individus de "haute qualité" que celle des individus de "basse qualité" - comporte généralement un aspect aléatoire.
- Chaque individu x_i se voit attribuer une probabilité p_i d'être choisi d'autant plus grande que son évaluation est haute (basse, dans le cas d'une sélection de "mauvais" individus).
- On tire un nombre r au hasard (uniformément) entre 0 et 1. L'individu k est choisi tel que :

$$\sum_{i=1}^{k-1} p_i < r \leq \sum_{i=1}^{k} p_i$$

- La procédure est itérée jusqu'à ce que l'on ait choisi un nombre fixé d'individus.

Croisement

Soit deux solutions x et y sélectionnées parmi les solutions de haute qualité. Un opérateur de croisement (crossover) fabrique une ou deux nouvelles solutions x', y' en combinant x et y.

Exemple 37 (Two-point crossover).

- x et y vecteurs 0-1 ;
- sélectionner aléatoirement deux positions dans les vecteurs et permuter les séquences de 0 et de l figurant entre ces deux positions dans les deux vecteurs.
- Pour les vecteurs :

$$\begin{array}{rcccccccc} x & = & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ y & = & 1 & 1 & 0 & 0 & 1 & 0 & 1 & O \end{array}$$

si les positions "après 2" et "après 5" sont choisies, on obtient :

$$\begin{array}{rcccc|ccc|ccc} x' & = & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ y' & = & 1 & 1 & 1 & 0 & 1 & 0 & 1 & O \end{array}$$

Mutation

- Une mutation est une perturbation introduite pour modifier une solution individuelle, par exemple la transformation d'un 0 en un 1 ou inversément dans un vecteur binaire.
- En général, l'opérateur de mutation est appliqué parcimonieusement : on décide de "muter" une solution avec une probabilité assez faible (de l'ordre de quelques centièmes, tout au plus).
- Un but possible de la mutation est d'introduire un élément de diversification, d'innovation comme dans la théorie darwinienne de l'évolution des espèces.

Remarques finales

- La recherche tabou peut être très efficace, mais implémentation et ajustement des paramètres difficiles, forts dépendants de la structure du problème.
- Les algorithmes génétiques sont efficaces si la structure du problème est bien exploitée.
- Méthodes hybrides très efficaces (exemple : recherche locale utilisée comme opérateur de mutation).