

Modélisation et Optimisation par les Graphes et la Programmation Linéaire

PATRICE PERNY

patrice.perny@lip6.fr

LIP6 – UPMC

Master Informatique – M1 – MOGPL – 2022-2023

Quelques informations utiles

Cours/TD

- Cours : Amphi 44, vendredi 8h30-10h30
- TD1 : Lundi 13h45-18h00 (14-24, 212)
- TD2 : Jeudi 13h45-18h00 (55-65, 101)
- TD3 : Vendredi 10h45-12h45 (24-34, 212) et 13h45-15h45 (Atrium, 521)
- TD4 : Vendredi 10h45-12h45 (Atrium, 259) et 13h45-15h45 (13-14, 102)
- Attention : pour les TDs 5 et 10 la deuxième partie sera un TME (voir salle sur planning)

Equipe pédagogique

- Cours : Patrice Perny, Evripidis Bampis
- TDs/TMEs : Evripidis Bampis, Nawal Benabbou, Bruno Escoffier, Patrice Perny.

Supports de cours et TDs

- Transparents du cours et compléments disponibles sur moodle (ajoutés au fur et à mesure de la progression du cours)
- Poly de sujets de TDs et TME (distribué en TD).
- Bibliographie : <http://androide.lip6.fr/?q=node/23>

Evaluation en MOGPL → 2 examens répartis + note projet(s)
poids des notes = (0.4, 0.4, 0.2)

2 / 41

Plan du cours (10 séances de 2 heures)

Partie I - Programmation linéaire

- ① Introduction à la modélisation, formulation d'un PL, résolution graphique.
- ② Algorithme du simplexe, présentation algébrique, méthode des tableaux
- ③ Méthodes d'initialisation. Algorithme du simplexe forme révisée.
Introduction à la Dualité.
- ④ Dualité en PL et applications, analyse post-optimales.
- ⑤ Branch and Bound, applications à la PLNE et la PL en 0-1.

Partie II - Graphes

chemins, ordonnancement, flot max, flot max à coup min, affectation,
problèmes de transport...

Cours 1 – Introduction à la modélisation, formulation d'un PL, résolution graphique

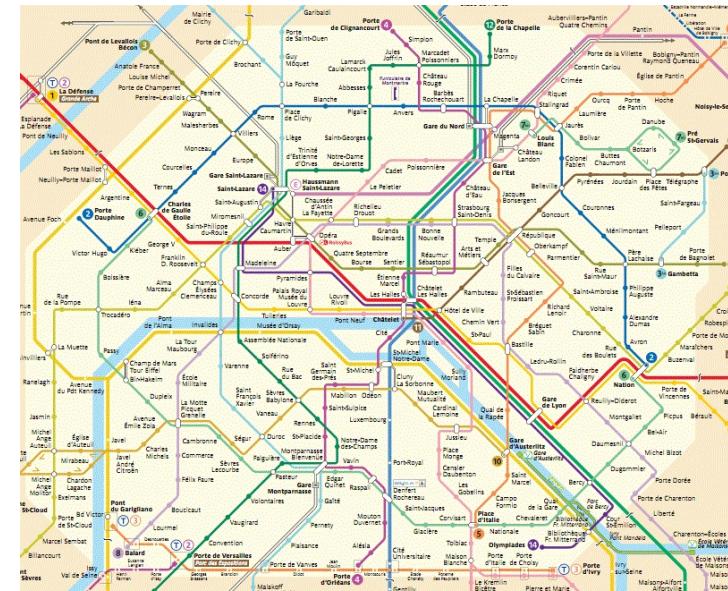
I) Introduction

Les graphes et la programmation linéaire (PL)

Les graphes

- intuitivement : points (sommets) et liens (arêtes, arcs)
relations binaires entre ces points
- un des outils formels les plus utilisés en informatique
- permet la représentation formelle de nombreux concepts, structures et problèmes du monde réel
- formalisme qui permet de faciliter la résolution de problèmes par la machine
- diverses librairies de programmes pour l'optimisation dans les graphs (e.g. python-graph, graph-tool en C++)
- exemples : metro, PERT, Jeux, Internet...

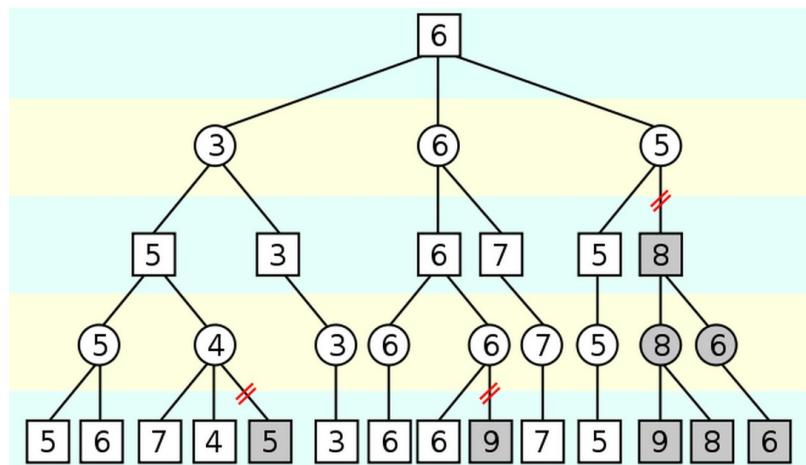
Exemple 1 : le graphe du Métro



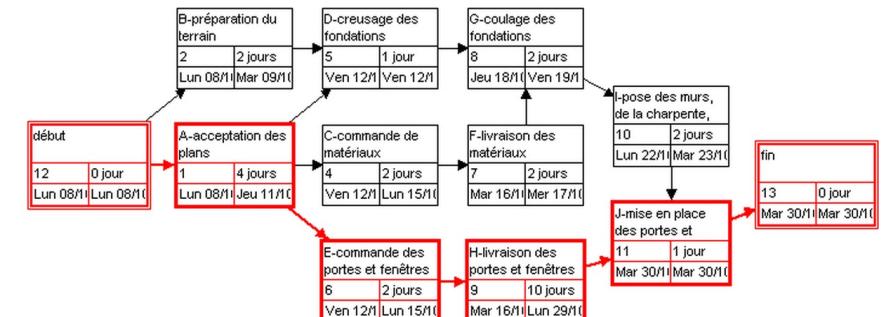
5 / 41

6 / 41

Exemple 2 : graphe d'un jeu



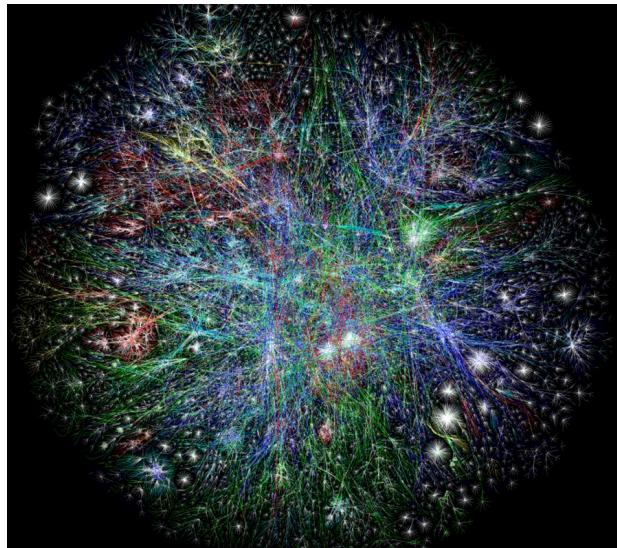
Exemple 3 : graphe PERT (orienté)



7 / 41

8 / 41

Exemple 4 : internet (grand graphe)



9 / 41

CPLEX : <https://www.ibm.com/fr-fr/analytics/cplex-optimizer>

IBM Analytics

Rechercher

CPLEX Optimizer

Solveur de programmation mathématique hautes performances pour programmation linéaire, programmation mixte en nombres entiers et programmation par contraintes

Essayer CPLEX Optimization Studio

Visite guidée de CPLEX Optimization Studio

PDF Référence client : Banque de France (18,5 Mo)

Présentation Avantages Produits Ressources Contacter un expert

11 / 41

La programmation linéaire

- optimisation d'une fonction linéaire sous contraintes linéaires

$$\begin{aligned} \max \text{ ou } \min \quad & c_1x_1 + c_2x_2 + \dots + c_nx_n \\ \left\{ \begin{array}{ll} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n & \leq b_i \quad i = 1, \dots, m_1 \\ a_{m+1,1}x_1 + a_{m+1,2}x_2 + \dots + a_{m+1,n}x_n & \geq b_i \quad i = m_1 + 1, \dots, m_2 \\ a_{m+2,1}x_1 + a_{m+2,2}x_2 + \dots + a_{m+2,n}x_n & = b_i \quad i = m_2 + 1, \dots, m \end{array} \right. \\ x_k \geq 0 \text{ si } k \in [1, n_1], \quad x_k \leq 0 \text{ si } k \in [n_1 + 1, n_2], \quad x_k \text{ qq si } k \in [n_2 + 1, n] \end{aligned}$$

- permet la représentation formelle de nombreux problèmes d'optimisation dans des nombreux domaines (e.g. finance, transports, communications, IA/RO)
- il existe aujourd'hui des solveurs très puissants CPLEX, Gurobi, GLPK qui résolvent des problèmes de taille importante
- utile aussi pour la résolution de problèmes combinatoires, de problème non linéaires etc
- entretient des liens étroits avec la théorie des graphes

10 / 41

Gurobi : <http://www.gurobi.com/>

Deutsch | Support | Contact Us

GUROBI OPTIMIZATION

Search website or documentation

PRODUCTS SUPPORT RESOURCES DOWNLOAD COMPANY

AN EASIER WAY TO BETTER DECISIONS

Gurobi Optimizer 5.6 - State-of-the-Art Mathematical Programming Solver

- Superior optimization algorithms for faster times to feasibility and optimality
- Flexible interfaces and modeling language support for maximum productivity
- Great support from easy-to-reach optimization experts
- Transparent pricing and flexible licensing so no surprises when it's time to deploy

GUROBI OPTIMIZER 5.6
High-end Optimization Libraries

ABOUT GUROBI
and why so many are choosing us

12 / 41

GLPK (GNU Linear Programming Kit)



[Introduction](#) | [Downloading](#) | [Documentation](#) | [Mailing Lists/Newsroups](#) | [Request an Enhancement](#) | [Report a bug](#)

Introduction to GLPK

The GLPK (GNU Linear Programming Kit) package is intended for solving large-scale linear programming (LP), mixed integer programming (M) written in ANSI C and organized in the form of a callable library.

GLPK supports the *GNU MathProg modeling language*, which is a subset of the AMPL language.

The GLPK package includes the following main components:

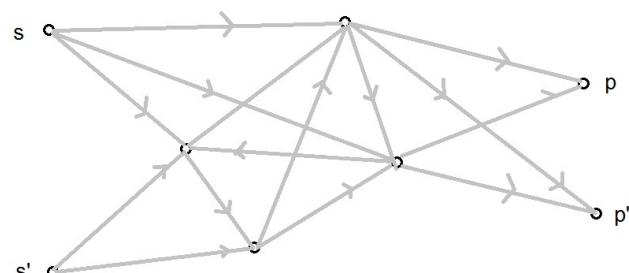
- primal and dual simplex methods
- primal-dual interior-point method
- branch-and-cut method
- translator for GNU MathProg
- application program interface (API)

13 / 41

14 / 41

Flots, Flots Max, Flots Max à coût min

RÉSEAU : graphe dont les arcs sont des possibilités de transférer de la matière d'un point à un autre



sources : s, s'

puits : p, p'

II) Un exemple de ce qui est fait en MOGPL

Interactions entre graphes et PL

Un problème d'allocation de bien/ressources indivisibles

Chaque individu i donne l'utilité $u_{ij} \in [0, 20]$ pour lui de l'objet j

| | | | | | |
|----|----|----|----|----|--|
| | | | | | |
| 16 | 12 | 18 | 8 | 2 | |
| 13 | 17 | 8 | 11 | 5 | |
| 20 | 0 | 16 | 4 | 2 | |
| 12 | 14 | 6 | 19 | 10 | |
| 5 | 8 | 3 | 15 | 18 | |



$$13 + 8 + 16 + 19 + 2 = 58$$

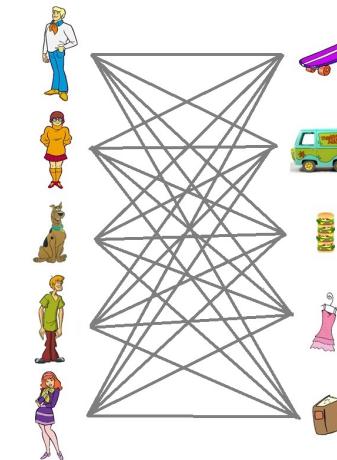


$$20 + 17 + 18 + 19 + 18 = 92$$

Différentes questions

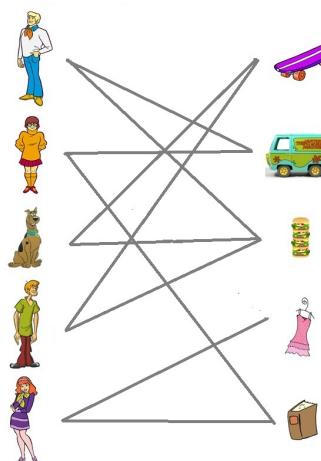
- ① Existe t-il une affectation telle que tous les individus aient une satisfaction $\geq \lambda$
- ② Trouver une affectation max-min optimale (solution équitable)
- ③ Trouver une affectation qui maximise la moyenne des satisfactions
- ④ Résoudre la question 3 avec la contrainte supplémentaire suivante : la satisf. moy des filles doit être \geq à la satisf. moy des garçons

Modélisation par un graphe bi-parti



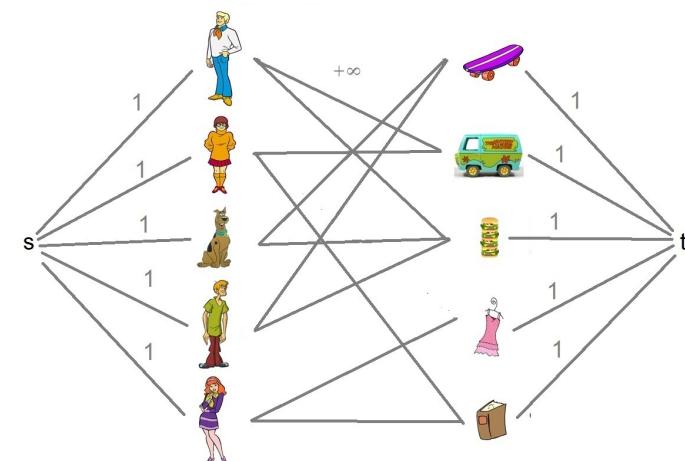
17 / 41

Question 1 : satisfaction minimale $\lambda = 13$



19 / 41

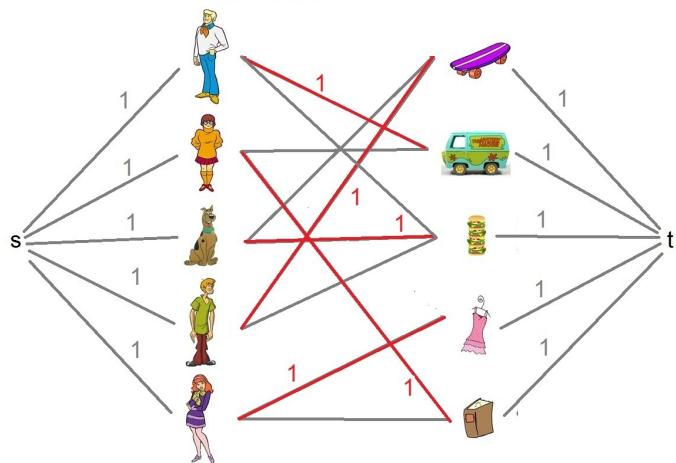
Question 1 : un problème de flot maximum



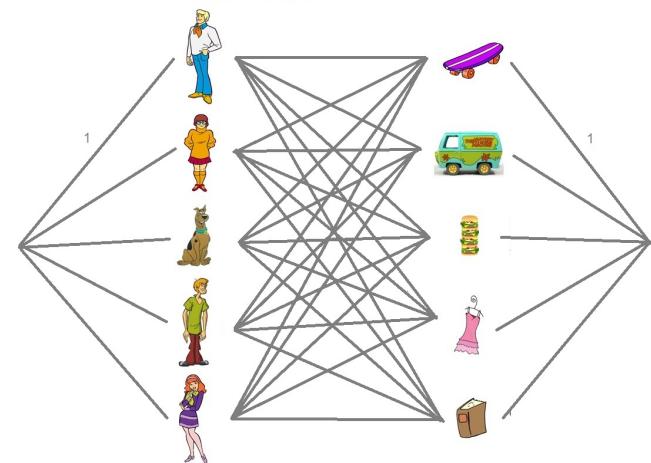
18 / 41

20 / 41

Question 1 et 2 : la solution optimale



Question 3 : un problème de flot max à coût min



21 / 41

22 / 41

Question 4 : modélisation par la PL(NE)

$$\max u(x) = \sum_{i=1}^n \sum_{j=1}^n u_{ij} x_{ij}$$

Formulation PL du problème d'affectation classique :

$$\text{s.c. } \begin{cases} \sum_{j=1}^n x_{ij} \leq 1 \\ \sum_{i=1}^n x_{ij} \leq 1 \end{cases}$$

$$x_{ij} \in [0, 1]$$

Pour Q4 : il suffit d'imposer $x_{ij} \in \{0, 1\}$ et d'ajouter la contrainte et :

$$\left[\sum_{i \in F} \sum_{j=1}^n u_{ij} x_{ij} \right] / |F| - \left[\sum_{i \in G} \sum_{j=1}^n u_{ij} x_{ij} \right] / |G| \geq 0$$

F : ensemble des filles, G : ensemble des garçons

III) Modélisation et résolution par la PL

Modélisation et bases de la résolution

23 / 41

24 / 41

Les étapes de la résolution d'un problème

- ① *la formulation du problème* : détermination de l'objectif, étude des contraintes
- ② *la modélisation* : construire un modèle mathématique représentant le plus fidèlement possible le problème
- ③ *l'élaboration d'une solution* : conception, développement d'un algorithme ou d'une méthode de résolution (ou utilisation d'un solveur) pour obtenir une solution optimale.
- ④ *analyse de sensibilité* : contrôle de la validité de la solution lors de modifications des paramètres

Une des spécificités de MOGPL : importance accordée à la phase 2, on ne se focalise pas seulement sur les algorithmes de la phase 3.

Modélisation d'un problème par un PL

Concerne les problèmes consistant à rechercher une décision optimale dans un système soumis à des contraintes linéaires.

- *identifier sur quoi porte la décision* : déterminer les inconnues sur lesquelles on a pouvoir de décision → variables de décision
- *identifier sur quoi portent les contraintes* : écriture des contraintes sur la forme de fonctions linéaires de variables de décision
- *identifier de quoi dépend l'objectif* : exprimer la fonction objectif sous forme d'une fonction linéaire des variables de décision (si elle n'est pas linéaire a priori, on pourra recourir à une linéarisation moyennant l'ajout de variables supplémentaires ; exemple de l'optimisation maxmin ou minmax).

25 / 41

26 / 41

Exemple de problème à modéliser

PROBLEME 1

Une raffinerie produit de l'essence ordinaire E_1 et du super E_2 à partir de 3 constituants de base B_1, B_2, B_3 dont les disponibilités journalières sont respectivement de 3000, 2000 et 1000 barils par jour. Pour des raisons de limitation de la pollution, un baril de super ne doit pas contenir plus de 30% de B_1 ; il ne peut pas contenir non plus moins de 40% de B_2 , ni moins de 50% de B_3 . De la même manière, un baril d'essence ordinaire ne peut contenir plus de 50% de B_1 ni moins de 10% de B_3 (pas de contrainte sur B_2).

Le prix d'achat des constituants de base est respectivement de 1200, 2400 et 2000 Euros par baril. Le prix de vente des carburants produits est de 1800 Euros par baril pour E_1 et 2200 Euros par baril pour E_2 .

On cherche la structure de production qui maximise la marge totale d'exploitation de la raffinerie sur la production des deux types de carburant (marge = bénéfice des ventes - coût d'achat des produits constituants).

Modélisation du problème (1/2)

Variables candidates :

- b_i : quantités de constituant de base B_i utilisée dans le mélange, $i = 1, 2, 3$, en barils (contraintes disponibilité, coût d'achat).
- e_j : quantité d'essence E_j produite $j = 1, 2$, en barils (bénéfice tiré des quantités vendues).
- x_{ij} : quantité de constituant de base B_i entrant dans la composition de E_j (en barils)

Tous ces variables sont utiles mais seules les x_{ij} sont vraiment nécessaires pour les contraintes de proportion qui s'écrivent :

$$\begin{aligned}x_{12} &\leq 0.3(x_{12} + x_{22} + x_{32}) \\x_{22} &\geq 0.4(x_{12} + x_{22} + x_{32}) \\x_{32} &\geq 0.5(x_{12} + x_{22} + x_{32}) \\x_{11} &\leq 0.5(x_{11} + x_{21} + x_{31}) \\x_{31} &\geq 0.1(x_{11} + x_{21} + x_{31})\end{aligned}$$

27 / 41

28 / 41

Modélisation du problème (2/2)

Les variables de décision sont les x_{ij} , les autres variables ne sont que des variables auxiliaires.

$$b_i = x_{i1} + x_{i2}, i = 1, 2, 3$$

$$e_j = x_{1j} + x_{2j} + x_{3j}, j = 1, 2$$

Expression de l'objectif

- Prix de revient : $12(x_{11} + x_{12}) + 24(x_{21} + x_{22}) + 20(x_{31} + x_{32})$
- Prix de vente : $18(x_{11} + x_{21} + x_{31}) + 22(x_{12} + x_{22} + x_{32})$
- Marge : $6x_{11} - 6x_{21} - 2x_{31} + 10x_{12} - 2x_{22} + 2x_{32}$

$$\text{donc } \max 6x_{11} - 6x_{21} - 2x_{31} + 10x_{12} - 2x_{22} + 2x_{32}$$

Formes canoniques et formes standards d'un PL

$$c^t = (c_1, \dots, c_n) \quad x^t = (x_1, \dots, x_n)$$

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_m \end{pmatrix}$$

Formes canoniques

$$\mathcal{P} \quad \begin{cases} \max z = c^t x \\ Ax \leq b \\ x \geq 0 \end{cases} \quad \mathcal{P}' \quad \begin{cases} \min z = c^t x \\ Ax \geq b \\ x \geq 0 \end{cases}$$

Formes standards

$$\mathcal{P} \quad \begin{cases} \max z = c^t x \\ Ax = b \\ x \geq 0 \end{cases} \quad \mathcal{P}' \quad \begin{cases} \min z = c^t x \\ Ax = b \\ x \geq 0 \end{cases}$$

29 / 41

30 / 41

Définitions

Définition

Soit \mathcal{P} un problème qui consiste à maximiser ou minimiser $c^t x$ sous les contraintes $Ax \leq b, x \geq 0$. On utilise la terminologie suivante :

- x : vecteur des variables de décision
- A matrice des contraintes, taille (m, n)
- b second membre
- c gradient de la fonction objectif

Définition (Solution réalisable)

$x \in \mathbb{R}^n$ est dite réalisable si elle satisfait à toutes les contraintes du problème, e.g. pour une forme canonique, $Ax \leq b, x \geq 0$.

Définition (Solution optimale)

$x \in \mathbb{R}^n$ est dite solution optimale si elle est réalisable et donne à la fonction objectif sa valeur optimale (max ou min selon les cas).

IV) Interprétation géométrique et résolution graphique d'un PL

31 / 41

32 / 41

Interprétation géométrique d'un PL

Définition (Hyperplan)

Un hyperplan de \mathbb{R}^n est un sous-espace affine de dimension $n - 1$.

Il a une équation de la forme : $a_1x_1 + a_2x_2 + \dots + a_nx_n = b$.

Définition (Demi-espace fermé)

Un demi-espace de \mathbb{R}^n est l'une des deux parties de \mathbb{R}^n délimitée par un hyperplan. Il est dit fermé s'il contient l'hyperplan. Un demi-espace fermé a une inéquation linéaire de la forme :

$$a_1x_1 + a_2x_2 + \dots + a_nx_n \leq b$$

Définition (Polyèdre convexe)

Un polyèdre convexe est une intersection finie de demi-espaces fermés. Il est caractérisé par un système d'inéquations linéaires.

REMARQUE : l'ens. des sol. réalisables d'un PL est un polyèdre convexe.

Face, arrête, sommet

Définition (Face du polyèdre)

Une face est l'intersection du polyèdre et de k hyperplans indépendants avec $k \leq n$.

Définition (Arête du polyèdre)

Face non-vide avec $k = n - 1$

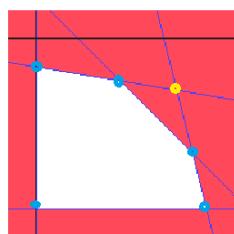
Définition (Sommet réalisable du polyèdre)

Face non-vide avec $k = n$

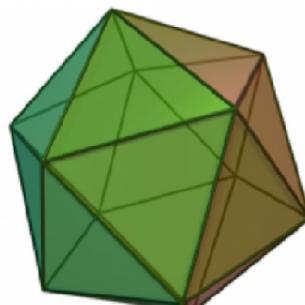
Définition (Sommet non-réalisable du polyèdre)

Intersection de n hyperplans séparateurs du polyèdre qui n'appartient pas au polyèdre

Exemple de polyèdres dans \mathbb{R}^2 et \mathbb{R}^3



Faces, arrêtes, sommets...



Résolution graphique d'un PL

Méthode applicable pour les problèmes admettant seulement deux variables de décision.

- ① Formuler le problème sous forme d'un PL.
- ② Dessiner les hyperplans (droites) associées aux contraintes du problème.
- ③ Pour chaque hyperplan, éliminer le ou les demi-espaces qui ne satisfont pas la contrainte.
- ④ Dessiner le vecteur gradient de la fonction objectif.
- ⑤ Déterminer un sommet optimal en utilisant les lignes de niveau perpendiculaires à la direction du gradient (adapter le sens de déplacement selon que l'on est en maximisation ou minimisation).
- ⑥ Déterminer les coordonnées du sommet et la valeur de l'objectif à l'optimum.

Un exemple de résolution graphique

PROBLEME 2

Une entreprise fabrique deux produits P_1, P_2 à l'aide de 3 matières premières M_1, M_2, M_3 . Les règles de compositions et prix de vente sont données ci-dessous (nb d'unités de M_i pour faire une unité de P_j) :

| | P_1 | P_2 | dispo. max |
|-------------|-----------|-----------|------------|
| M_1 | 1 | 6 | 30 |
| M_2 | 2 | 2 | 15 |
| M_3 | 4 | 1 | 24 |
| Px de vente | 2 Euros/u | 3 Euros/u | |

Modélisation du problème 2

Variables de décision

x_1 : quantité mensuelle de produit P_1 fabriqué

x_2 : quantité mensuelle de produit P_2 fabriqué

Programme Linéaire

$$\max z = 2x_1 + 3x_2$$

$$\mathcal{P} \quad \begin{cases} x_1 + 6x_2 \leq 30 \\ 2x_1 + 2x_2 \leq 15 \\ 4x_1 + x_2 \leq 24 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

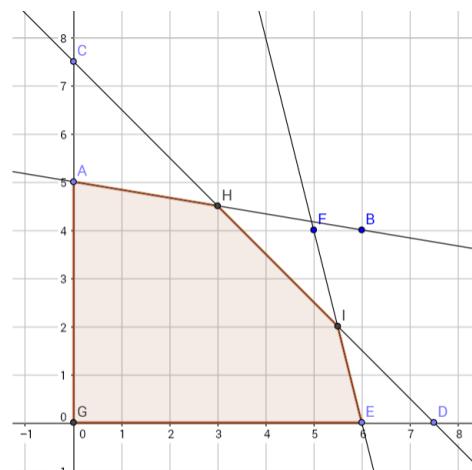
Résolution graphique (voir figures pages suivantes).

Solution optimale

$$x_1^* = 3, x_2^* = 9/2 \text{ et } z^* = 39/2$$

Représentation des contraintes

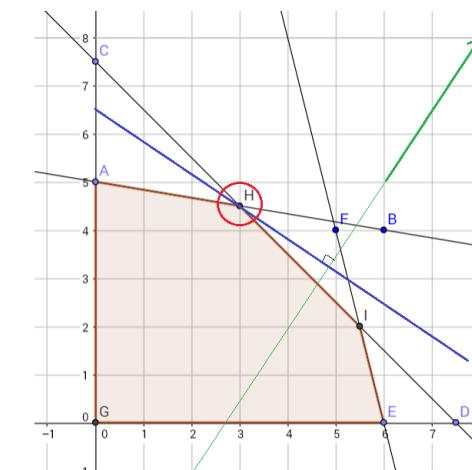
$$\begin{aligned} x_1 + 6x_2 &= 30 & (0, 5) & (6, 4) \\ 2x_1 + 2x_2 &= 15 & (\frac{15}{2}, 0) & (0, \frac{15}{2}) \\ 4x_1 + x_2 &= 24 & (6, 0) & (5, 4) \end{aligned}$$



37 / 41

Détermination de la solution optimale

$$\begin{aligned} x_1 + 6x_2 &= 30 & (0, 5) & (6, 4) \\ 2x_1 + 2x_2 &= 15 & (\frac{15}{2}, 0) & (0, \frac{15}{2}) \\ 4x_1 + x_2 &= 24 & (6, 0) & (5, 4) \end{aligned}$$



39 / 41

38 / 41

40 / 41

A réviser avant le prochain cours

Quelques notions de base en algèbre linéaire...

- espace vectoriel sur \mathbb{R}
- produit matriciel, transposition,
- matrice d'une application linéaire,
- famille libre, famille liée, famille génératrice, base
- résolution de systèmes d'équations linéaires
- inversion de matrice
- changement de base, matrice de passage