Graphes et Recherche Opérationnelle

Introduction générale

J.-F. Scheid, B. Pinçon

2024-2025

Fonctionnement du module GRO I

Evaluation (CC) :

- un test écrit (ou qcm sur machine) sur la partie 1 (autour de 30 mn) ;
- un TP noté (1h50) ;
- un examen (sur tout le programme) (environ 2h00) avec une partie test (sans document) sur la partie 2 (autour de 30 mn).
- $NF = \dfrac{Test \ + \ 3 \times TP \ + \ 3 \times E}{7}$

Documents : site Arche GRO (voir à la fin pour quelques livres).

TPs : 3 TPs de 2h en python (certains TPs utilisent le solveur GUROBI à partir de python). **(avoir une version de python avec numpy, scipy, matplotlib, opérationnelle sur votre machine.)**

Fonctionnement du module GRO II

Contenu du cours.

- *Programmation linéaire*: modélisation, aspects géométriques, méthode du simplexe, analyse de sensibilité, dualité ?
- *Graphes et R.O* :
  - *Flot optimal dans un graphe*
  - *Programmation linéaire en nombres entiers (PLNE)*
  - *Introduction aux méthodes heuristiques* ? : algorithme A^*, recuit simulé, algorithmes génétiques, colonies de fourmis.

Introduction générale

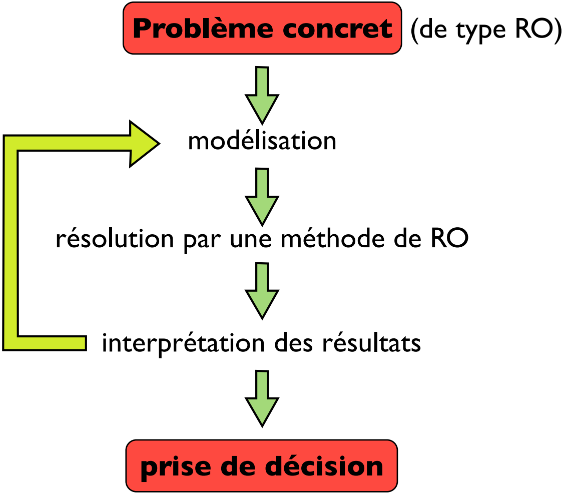
Le terme *recherche opérationnelle* (Operations Research en anglais) date de la seconde guerre mondiale où les anglo-saxons ont cherché à optimiser l'implantation de radars de défense anti-aérienne et à déterminer la taille optimale des convois d'approvisionnement.

Une définition possible de la RO:

Ensemble de méthodes d'analyse scientifique (maths et info) des phénomènes d'organisation qui traite de l'**optimisation** de l'architecture et du fonctionnement des systèmes (industriels, économiques, numériques ...).

La RO est un outil d'**aide à la décision**.

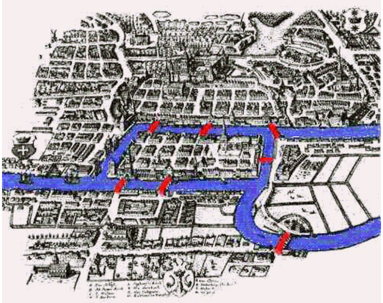
Schéma général:



Quelques exemples et domaines d'applications de la RO

Quelques exemples et domaines d'applications de la RO

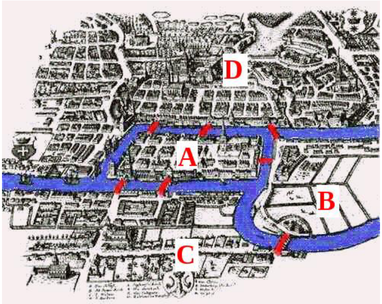
1. Un premier exemple : *Les ponts de Königsberg* (Euler 1735).



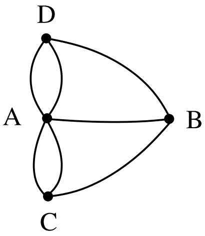
Deux îles reliées entre elles par 1 pont et 6 ponts relient les îles à la ville.

A partir d'un point de départ quelconque, existe-t-il un chemin passant une et une seule fois par chaque pont et qui revient au point de départ ?

La réponse est **NON**. La preuve utilise un **graphe**: les arêtes du graphe représentent les ponts et les sommets les différents endroits de la ville (rives gauche et droite, les deux îles).



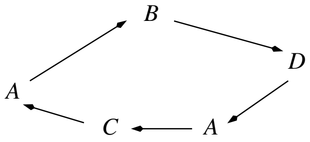
$\Longrightarrow$



Si la réponse était OUI alors il y aurait nombre pair d'arêtes reliées à chaque noeud $\rightarrow$ contradiction (voir page suivante).

Introduction générale

Preuve (idée): Supposons qu'il existe un tel chemin, par exemple[1] :

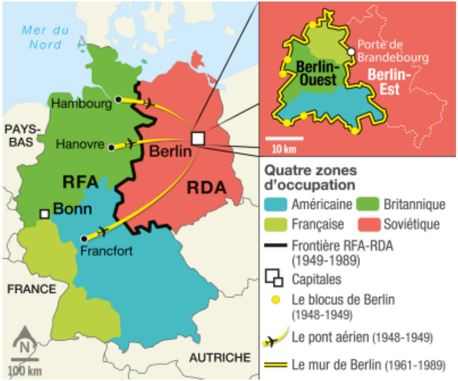


Alors il est clair que le nombre d'arêtes (de ponts) relié un noeud quelconque doit être pair. Ici sur cet exemple, on aurait donc 4 arêtes reliant A et deux pour les autres noeuds.

[1] Ce chemin ne fonctionne pas car le pont entre C et B n'est pas traversé de même que l'un des deux ponts entre A et D.

2. *Programmation linéaire pour l'organisation optimale de convois (G. Dantzig, 1947)*.

Blocus de Berlin-Ouest en juin 1948 (15 mois).



Convois d'avions US et anglais pour approvisionner les 2 millions de personnes de Berlin-Ouest.

1 avion toutes les 45s au plus fort du trafic avec 278000 vols en 15 mois.

Introduction générale

Variables : avions US et anglais, équipages, pistes, provisions, dépenses.

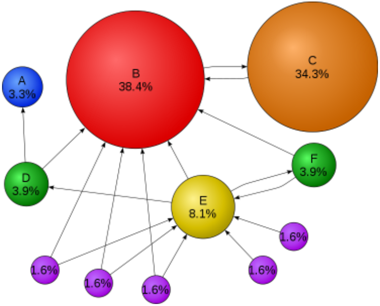
Contraintes :

- les avions US sont plus gros et ils nécessitent 2 fois plus de personnels par avion (1 équipage par avion anglais contre 2 pour un avion US).
- 64 équipages, 44 avions disponibles.
- un vol US coûte 9000$ contre 5000$ pour un vol anglais. Le coût total par semaine est limité à 300.000$
- un avion US a une capacité de $850m^3$ contre $560m^3$ pour un avion anglais.
- $\cdots$

Objectif : déterminer le nombre d'avions US et anglais à utiliser pour maximiser le contenu total embarqué.

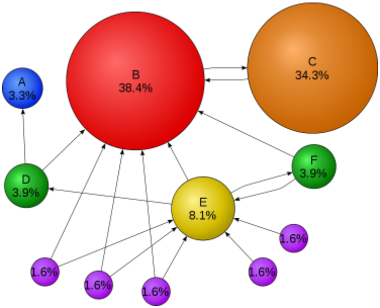
3. *Graphe et PageRank (Larry Page - Sergey Brin, 1998)*.
   Classement des pages Web utilisé par les moteurs de recherches.

   Modélisation du Web : graphe orienté dont les sommets j représentent les pages du Web et les arcs $j \to i$ les hyperliens.



3. *Graphe et PageRank (Larry Page - Sergey Brin, 1998)*.
   Classement des pages Web utilisé par les moteurs de recherches.

   Modélisation du Web : graphe orienté dont les sommets j représentent les pages du Web et les arcs $j \to i$ les hyperliens.



Principe du PageRank: *une page i est importante si beaucoup de pages importantes pointent vers i.*

Importance de la page i :

$$\mu_i = \sum_{j \to i} \frac{1}{l_j} \mu_j$$

l_j : nb de liens sortant de la page j

$\Rightarrow$ pb aux valeurs propres $\mu = A\mu$.

4. *Problème de production*.

   Détermination d'un plan optimal de fabrication: une entreprise fabrique plusieurs produits, ce qui exige des ressources particulières (matières premières, machines, personnel ...) en quantités limitées. Chaque produit rapporte un certain bénéfice (connu) à l'entreprise.

   Quels produits l'entreprise doit-elle fabriquer et en quelle quantité pour réaliser le bénéfice total le plus élevé ?

4. *Problème de production*.

Détermination d'un plan optimal de fabrication: une entreprise fabrique plusieurs produits, ce qui exige des ressources particulières (matières premières, machines, personnel ...) en quantités limitées. Chaque produit rapporte un certain bénéfice (connu) à l'entreprise.

Quels produits l'entreprise doit-elle fabriquer et en quelle quantité pour réaliser le bénéfice total le plus élevé ?

$\rightarrow$ modélisation par programmation linéaire (simplexe).

x_i : quantité (non entière) du produit i fabriqué.

$$\max_{\mathbf{x}} F(\mathbf{x}) = \mathbf{c}^\top \mathbf{x} = c_1 x_1 + \cdots c_n x_n \;\leftarrow \text{l'objectif}$$
$$\begin{cases} A\mathbf{x} \leq \mathbf{b} \;\leftarrow \text{les contraintes} \\ \mathbf{x} \geq 0 \end{cases}$$

Un exemple de problème de production.

Une usine fabrique 2 produits P_1 et P_2 nécessitant des ressources d'équipement, de main d'oeuvre et de matières premières disponibles en quantité limitée.

	P_1	P_2	disponibilité
équipement	3	9	81
main d'œuvre	4	5	55
matière première	2	1	20

P_1 et P_2 rapportent à la vente 6 euros et 4 euros par unité.

Un exemple de problème de production.

Une usine fabrique 2 produits P_1 et P_2 nécessitant des ressources d'équipement, de main d'oeuvre et de matières premières disponibles en quantité limitée.

	P_1	P_2	disponibilité
équipement	3	9	81
main d'œuvre	4	5	55
matière première	2	1	20

P_1 et P_2 rapportent à la vente 6 euros et 4 euros par unité.

Quelles quantités (non entières) de produits P_1 et P_2 doit produire l'usine pour maximiser le bénéfice total venant de la vente des 2 produits ?

Introduction générale

x_1, x_2 sont les quantités des produits P_1 et P_2 fabriqués $(x_1, x_2 \in \mathbb{R})$

- *Objectif*

  Bénéfice total $F(x_1, x_2) = 6x_1 + 4x_2$.

  On cherche

$$\max_{(x_1,x_2)} \left[F(x_1, x_2) = 6x_1 + 4x_2\right].$$

x_1, x_2 sont les quantités des produits P_1 et P_2 fabriqués $(x_1, x_2 \in \mathbb{R})$

- *Objectif*

  Bénéfice total $F(x_1, x_2) = 6x_1 + 4x_2$.

  On cherche

  $$\max_{(x_1,x_2)} \left[F(x_1, x_2) = 6x_1 + 4x_2\right].$$

- *Contraintes*
  - Disponibilité de chacune des ressources :

    $$\begin{array}{ll} 3x_1 + 9x_2 \leq 81 & \text{(équipement)} \\ 4x_1 + 5x_2 \leq 55 & \text{(main d'oeuvre)} \\ 2x_1 + x_2 \leq 20 & \text{(matière première)} \end{array}$$

  - Positivité des variables: $x_1, x_2 \geq 0$.

5. *Problème de transport/logistique*.

Une entreprise dispose de plusieurs entrepôts contenant chacun un certain nombre de containers. Plusieurs magasins commandent des containers. On connaît le coût de transport de chaque dépôt aux magasins.

Quelle est l'organisation des livraisons des containers pour minimiser le coût total de transport ?

Modélisation de problèmes de transport/logistique par programmation linéaire **en nombres entiers** / "Branch-and-Bound"

Un exemple. 3 magasins (M_1,M_2,M_3) et deux entrepôts (E_1,E_2).

Tableau des coûts :

	M_1	M_2	M_3	disponibilité entrepôts $\downarrow$
E_1	5	3	4	8
E_2	6	7	2	9
demande magasins $\rightarrow$	4	5	8	

Quelle est l'organisation des livraisons des containers pour minimiser le coût total de transport ?

x_{ij} : nombre de containers de l'entrepôt E_i acheminés au magasin M_j ($1 \leq i \leq 2$ et $1 \leq j \leq 3$.)

$$min_{\mathbf{x}} \left(F(\mathbf{x}) = 5x_{11} + 3x_{12} + 4x_{13} + 6x_{21} + 7x_{22} + 2x_{23}\right)$$

$$\begin{array}{ll}
x_{11} + x_{21} = 4 & \text{(demande du magasin } M_1\text{)} \\
x_{12} + x_{22} = 5 & \text{(demande du magasin } M_2\text{)} \\
x_{13} + x_{23} = 8 & \text{(demande du magasin } M_3\text{)} \\
x_{11} + x_{12} + x_{13} \leq 8 & \text{(disponibilité entrepôt } E_1\text{)} \\
x_{21} + x_{22} + x_{23} \leq 9 & \text{(disponibilité entrepôt } E_2\text{)}
\end{array}$$

$$x_{ij} \ \textbf{entiers}, \forall i, j$$

5. *Problèmes d'affectation*.

   n tâches doivent être affectées à n machines (1 tâche par machine). Le coût d'exécution de chaque tâche par chacune des machines est connu.

   Trouver l'affectation qui minimise le coût total.

   $\rightarrow$ programmation linéaire en **variables binaires** / flot maximal à travers un graphe

a) un premier exemple de problème d'affectation en variables binaires *: affectation des cours d'ouverture du Collégium INP de l'UL*

Affectations des cours d'ouverture en 2014-15 par programmation linéaire (collaboration avec Telecom Nancy Services):

1578 élèves, 78 cours proposés sur 3 jours (avec répétitions) .

$\rightarrow$ Plus de 150 000 variables :

$$x_{ij} = \begin{cases} 1 & \text{si l'élève } i \text{ suit le cours } j \\ 0 & \text{sinon} \end{cases}$$

Introduction générale

Chaque élève doit suivre 3 cours et donne une liste de préférence d'au moins 6 cours.

$\rightarrow$ maximisation de la satisfaction collective (globale) :

$$\max_{x_{ij}} \sum_{i,j} c_{ij} x_{ij}$$

où $c_{ij} = \begin{cases} 1 & \text{si l'élève } i \text{ a choisi le cours } j \text{ dans sa liste de préférence} \\ 0 & \text{sinon} \end{cases}$

Introduction générale

Chaque élève doit suivre 3 cours et donne une liste de préférence d'au moins 6 cours.

→ maximisation de la satisfaction collective (globale) :

$$\max_{x_{ij}} \sum_{i,j} c_{ij} x_{ij}$$

où $c_{ij} = \begin{cases} 1 & \text{si l'élève } i \text{ a choisi le cours } j \text{ dans sa liste de préférence} \\ 0 & \text{sinon} \end{cases}$

... et plus de 43 000 contraintes !

→ contraintes linéaires sur les variables de type égalités/inégalités.
Par exemple : (chaque élève suit exactement 3 cours)

$$\sum_j x_{ij} = 3, \quad \text{pour tout } i$$

Introduction générale

Solveur de programmation linéaire en nombres entiers CPLEX (IBM) :

$<$ 2 min sur station de travail Intel Xeon 2.26 GHz (4 cores/8 threads), 24GB RAM.

Taux de satisfaction $>$ 97%.

b) un autre exemple de problème d'affectation : le sudoku

2	·	·	·	·	4	·	3	·
·	9	·	·	·	·	·	7	·
·	·	·	·	6	·	·	4	·
3	8	·	·	·	7	·	·	·
6	·	·	·	·	·	5	·	·
·	·	2	·	·	·	8	·	·
·	·	·	8	·	·	9	·	·
·	·	·	·	9	·	·	·	2
·	·	4	·	·	2	·	8	·

b) un autre exemple de problème d'affectation : le sudoku

2	·	·	·	·	**4**	·	**3**	·
·	**9**	·	·	·	·	·	**7**	·
·	·	·	·	**6**	·	·	**4**	·
3	**8**	·	·	·	**7**	·	·	·
6	·	·	·	·	·	**5**	·	·
·	·	**2**	·	·	·	**8**	·	·
·	·	·	**8**	·	·	**9**	·	·
·	·	·	·	**9**	·	·	·	**2**
·	·	**4**	·	·	**2**	·	**8**	·

2	1	5	7	8	**4**	6	**3**	9
4	**9**	6	2	5	3	1	**7**	8
8	3	7	1	**6**	9	2	**4**	5
3	**8**	9	5	2	**7**	4	1	6
6	7	1	9	4	8	**5**	2	3
5	4	**2**	3	1	6	**8**	9	7
1	2	3	**8**	7	5	**9**	6	4
7	6	8	4	**9**	1	3	5	**2**
9	5	**4**	6	3	**2**	7	**8**	1

b) un autre exemple de problème d'affectation : le sudoku

2	·	·	·	·	**4**	·	**3**	·
·	**9**	·	·	·	·	·	**7**	·
·	·	·	·	**6**	·	·	**4**	·
3	**8**	·	·	·	**7**	·	·	·
6	·	·	·	·	·	**5**	·	·
·	·	**2**	·	·	·	**8**	·	·
·	·	·	**8**	·	·	**9**	·	·
·	·	·	·	**9**	·	·	·	**2**
·	·	**4**	·	·	**2**	·	**8**	·

2	1	5	7	8	**4**	6	**3**	9
4	**9**	6	2	5	3	1	**7**	8
8	3	7	1	**6**	9	2	**4**	5
3	**8**	9	5	2	**7**	4	1	6
6	7	1	9	4	8	**5**	2	3
5	4	**2**	3	1	6	**8**	9	7
1	2	3	**8**	7	5	**9**	6	4
7	6	8	4	**9**	1	3	5	**2**
9	5	**4**	6	3	**2**	7	**8**	1

Variables d'affectation : pour une case $(i, j) \in [\![1, 9]\!]^2$,

$$x_{ijk} = \begin{cases} 1 & \text{si le chiffre } k \in [\![1, 9]\!] \text{ est affecté à la case } (i, j) \\ 0 & \text{sinon} \end{cases}$$

$\rightarrow$ maximisation de la fonction $f \equiv 1$ avec contraintes sur les x_{ijk}.

c) un dernier exemple de problème d'affectation : *mariages stables*

Trouver n mariages stables entre hommes et femmes.

Il y a *instabilité* d'une situation (cad de n mariages formés) si un homme et une femme préfèreraient se mettre en couple plutôt que de rester avec leurs partenaires actuels.

c) un dernier exemple de problème d'affectation : *mariages stables*

Trouver n mariages stables entre hommes et femmes.

Il y a *instabilité* d'une situation (cad de n mariages formés) si un homme et une femme préfèreraient se mettre en couple plutôt que de rester avec leurs partenaires actuels.

Chaque homme et chaque femme exprime ses préférences parmi l'ensemble des individus de l'autre sexe (liste).

Un exemple (1=premier choix)

	Y_1	Y_2	Y_3
X_1	$(2,2)$	$(3,3)$	$(1,1)$
X_2	$(1,1)$	$(3,1)$	$(2,2)$
X_3	$(3,3)$	$(2,2)$	$(1,3)$

- 6 mariages possibles.
- (X_1, Y_3), (X_2, Y_1), (X_3, Y_3) est stable.
- (X_1, Y_2), (X_2, Y_3), (X_3, Y_1) est instable.

Un exemple (1=premier choix)

	Y_1	Y_2	Y_3
X_1	$(2,2)$	$(3,3)$	$(1,1)$
X_2	$(1,1)$	$(3,1)$	$(2,2)$
X_3	$(3,3)$	$(2,2)$	$(1,3)$

- 6 mariages possibles.
- (X_1, Y_3), (X_2, Y_1), (X_3, Y_3) est stable.
- (X_1, Y_2), (X_2, Y_3), (X_3, Y_1) est instable.

Propriété : il existe (au moins) une solution stable, algorithme de Gale-Shapley (1962).

Introduction générale

Quelques applications du problème des mariages stables:

- Services Internet distribués (page web, vidéos,...) avec utilisateurs/serveurs.
  - Les utilisateurs préfèrent accéder à des serveurs de proximité pour des réponses rapides (liste de préférences des serveurs pour chaque utilisateur).
  - Les serveurs préfèrent servir des utilisateurs dont le coût est faible (liste de préférences des utilisateurs pour chaque serveur).
- APB (jusqu'en 2017) puis Parcours-Sup (h=formation, f=étudiant)

Voir la page Wikipédia :

`https://fr.wikipedia.org/wiki/Problème_des_mariages_stables`

6. *Le problème de voyageur de commerce*.

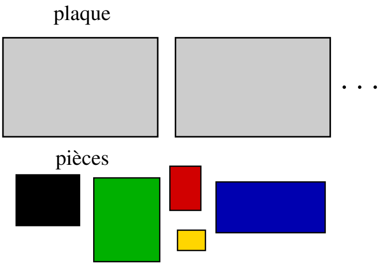
   Un voyageur de commerce doit visiter n villes. La distance entre chaque ville est donnée.

   Trouver le plus court trajet passant par les n villes.

   Problème très difficile. On peut le résoudre assez facilement (assez peu de lignes de code) en utilisant un principe de programmation dynamique : on décompose le problème en sous-problèmes et on résout les sous-problèmes des plus petits aux plus grands en stockant les résultats intermédiaires. On peut aussi le poser comme un programme linéaire. Il existe aussi beaucoup d'heuristiques.

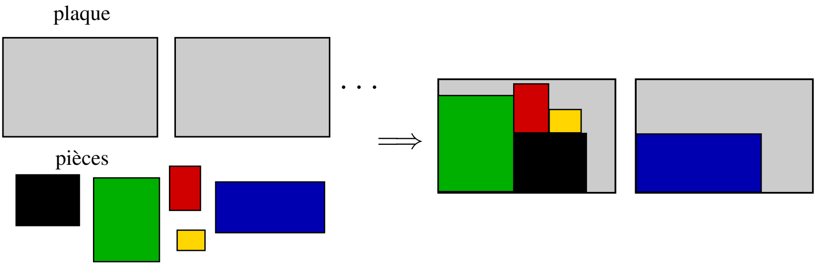
7. *Placement optimal de pièces 2D/3D (Bin Packing)*.

En 2D : on dispose de plaques rectangulaires toutes identiques dans lesquelles on veut placer des pièces rectangulaires sans chevauchement. Les pièces à placer ont des dimensions différentes.



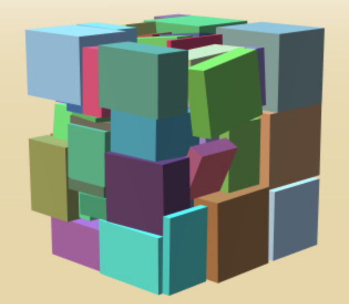
7. *Placement optimal de pièces 2D/3D (Bin Packing)*.

En 2D : on dispose de plaques rectangulaires toutes identiques dans lesquelles on veut placer des pièces rectangulaires sans chevauchement. Les pièces à placer ont des dimensions différentes.



Trouver le placement optimal des pièces pour minimiser le nombre de plaques utilisées.

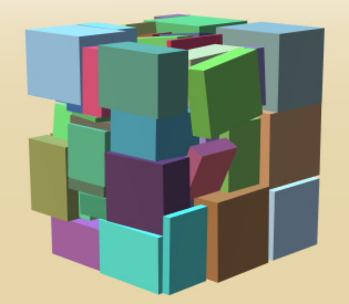
Aussi en **3D** ...



Daniel Hartmeier/ benzedrine.ch

Introduction générale

Aussi en **3D** ...



Daniel Hartmeier/ benzedrine.ch

→ méthodes heuristiques (du grec *eurisko*, "je trouve" : art de faire des découvertes, d'inventer) : fournir rapidement une solution réalisable mais pas nécessairement optimale.

- *algorithmes gloutons* (greedy algorithm) : faire un choix *localement* optimal
- *méthodes évolutionnistes* : algos génétiques, colonies de fourmis, ...

Tous les problèmes précédents sont des *problèmes de combinatoire*. Par ex., pour le problème du voyageur de commerce, il y a $n!$ possibilités.

Avec $n = 20$, l'énumération des trajets possibles à une vitesse d'un million par seconde, prendrait 77094 années ...

$\rightsquigarrow$ *trouver des méthodes efficaces (exactes, approchées ...)*.

Formalisation.

Presque tous les problèmes mentionnés ci-dessus et abordés dans les TD peuvent se formaliser de la façon suivante[2] :

$$\max \text{ ou } \min \{f(x),\ x \in X\}$$

- $f : X \to \mathbb{R}$ est la fonction **objectif** qui peut être linéaire, quadratique, nonlinéaire...
- X est l'ensemble des solutions possibles dites **réalisables** (les contraintes). L'ens. X est soit infini non dénombrable (pas de contraintes d'intégrité sur les variables) soit fini (les variables doivent être entières, voire binaires) mais en général de très grande taille. Il y a aussi des problèmes mixtes où seulement une partie des variables doivent être entière ou binaire.

[2] Pour un problème de *minimisation*, on se ramène à une *maximisation* avec $\min f = -\max(-f)$

Quelques ouvrages de références.

1. *Précis de recherche opérationnelle – Méthodes et exercices d'application*, Faure, Lemaire, Picouleau; 2008.
   → ouvrage de référence, un "classique".
2. *Exercices et problèmes résolus de recherche opérationnelle : Tome 1 (Graphes : leurs usages, leurs algorithmes), Tome 3 (Programmation linéaire et extensions, problèmes classiques)*. Roseaux : Billionnet, Carlier, Chrétienne, Lemaire, Faure; 1985/1998.
   → contient des exercices classiques en lien avec [1].
3. *Linear programming 1: Introduction*, G. Dantzig, M. Thapa, 1997. *Linear programming 2: Theory and Extensions*, G. Dantzig, M. Thapa, 2003.
   → ouvrage très complet.

Quelques ouvrages de références (suite).

4. *Optimisation discrète – De la modélisation à la résolution par des logiciels de programmation mathématique*, Billionnet, 2007.
   $\rightarrow$ ouvrage récent qui contient un exposé de méthodes plus modernes que [1]. Ouvrage intéressant.
5. *Recherche opérationnelle pour ingénieurs Tome I et II*, Hêche, Liebling, de Werra; 2003.
   $\rightarrow$ malgré son titre (!), contenu assez mathématique; contient des démonstrations intéressantes.
6. *Introduction à l'algorithmique*, Corman, Leiserson, Rivest, 2002.
   $\rightarrow$ aspects algorithmiques des maths numériques, de la Recherche Opérationnelle et des probabilités.