Introduction à l'optimisation

Stéphane Canu
stephane.canu@litislab.eu

ASI 4 - Programmation linéaire et le simplexe

December 1, 2016

Plan

Un exemple de programmation linéaire

Calculez le mélange le moins cher à donner à nos cobayes

Afin d'assurer une bonne santé de cobayes, il faut les nourrir en leur donnant un minimum de 24 grammes de lipides, 36 grammes de glucides et 4 grammes de protéines par jour. Il ne faut pas leur donner plus de 5 onces de nourriture.
Nous disposions de deux sources d'alimentation. Les croquettes royal cobaye qui contiennent 8 grammes de lipides, 12 grammes de glucides et 2 grammes de protéines par once et qui coutent 3 euros les 10 onces, et les boulettes « vite se casse » qui contiennent 12 grammes de lipides, 12 grammes de glucides et 1 grammes de protéines par once et qui coutent 2 euros les 10 onces.

Un exemple de programmation linéaire

Calculez le mélange le moins cher à donner à nos cobayes

Afin d'assurer une bonne santé de cobayes, il faut les nourrir en leur donnant un minimum de 24 grammes de lipides, 36 grammes de glucides et 4 grammes de protéines par jour. Il ne faut pas leur donner plus de 5 onces de nourriture.
Nous disposions de deux sources d'alimentation. Les croquettes royal cobaye qui contiennent 8 grammes de lipides, 12 grammes de glucides et 2 grammes de protéines par once et qui coutent 3 euros les 10 onces, et les boulettes « vite se casse » qui contiennent 12 grammes de lipides, 12 grammes de glucides et 1 grammes de protéines par once et qui coutent 2 euros les 10 onces.

inconnues

- quantité de royal cobaye x_1 en once
- quantité de « vite se casse » x_2 en once

cout minimiser par rapport à $\mathbf{x} = (x_1, x_2)$ la fonction cout $J(\mathbf{x}) = 30x_1 + 20x_2$

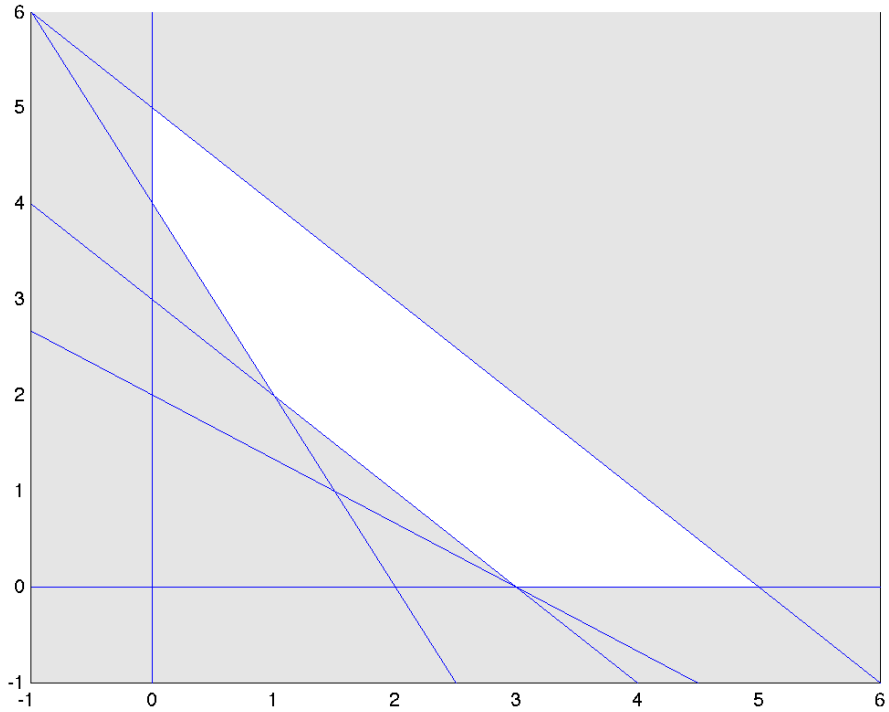
contraintes

- lipides $8x_1 + 12x_2 \geq 24$
- glucides $12x_1 + 12x_2 \geq 36$
- protéines $2x_1 + x_2 \geq 4$
- diététique $x_1 + x_2 \leq 5$
- ces quantités sont positives : $x_1 \geq 0$ et $x_2 \geq 0$

Domaine admissible

Contraintes :

- lipides $8x_1 + 12x_2 \geq 24$
- glucides $12x_1 + 12x_2 \geq 36$
- protéines $2x_1 + x_2 \geq 4$
- diététique $x_1 + x_2 \leq 5$
- ces quantités sont positives : $x_1 \geq 0$ et $x_2 \geq 0$



Cout

$$J(\mathbf{x}) = 30x_1 + 20x_2$$

Programmation linéaire avec Matlab

Contraintes :

- lipides $8x_1 + 12x_2 \geq 24$
- glucides $12x_1 + 12x_2 \geq 36$
- protéines $2x_1 + x_2 \geq 4$
- diététique $x_1 + x_2 \leq 5$
- ces quantités sont positives :

  $x_1 \geq 0$ et $x_2 \geq 0$

Cout

$$J(\mathbf{x}) = 30x_1 + 20x_2$$

CVX: Matlab Software for Disciplined Convex Programming
Version 2.0 (beta), November 2012, Build 883

cvxr.com/cvx/

```
cvx_begin
    variables x_1 x_2
    dual variables lip glu pro die p1 p2;
    minimize( 30*x_1 +  20*x_2 )
    subject to
         lip : 8 *x_1 + 12* x_2 >= 24;
         glu : 12 *x_1 + 12 *x_2 >= 36;
         pro : 2 *x_1 +  x_2 >= 4;
         die : x_1 +  x_2 <= 5;
         p1 : x_1 >=0;
         p2 : x_2 >=0;
cvx_end
```

Programmation linéaire - version matricielle

Cout avec

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \text{ et } \mathbf{f} = \begin{pmatrix} 30 \\ 20 \end{pmatrix}$$

$$\begin{aligned} J(\mathbf{x}) &= 30x_1 + 20x_2 \\ &= \mathbf{f}^\top \mathbf{x} \end{aligned}$$

Contraintes avec

$$\mathbf{b} = \begin{pmatrix} 24 \\ 36 \\ 4 \\ -5 \end{pmatrix} \text{ et } A = \begin{pmatrix} 8 & 12 \\ 12 & 12 \\ 2 & 1 \\ -1 & -1 \end{pmatrix}$$

lipides $8x_1 + 12x_2 \geq 24$
glucides $12x_1 + 12x_2 \geq 36$
protéines $2x_1 + x_2 \geq 4$
diététique $x_1 + x_2 \leq 5$

$\rightarrow A\mathbf{x} \geq \mathbf{b}$

```
cvx_begin
    variables x_1 x_2
    dual variables lip glu pro die p1 p2;
    minimize( 30*x_1 +  20*x_2 )
    subject to
         lip : 8 *x_1 + 12* x_2 >= 24;
         glu : 12 *x_1 + 12 *x_2 >= 36;
         pro : 2 *x_1 +  x_2 >= 4;
         die : x_1 +  x_2 <= 5;
         p1 : x_1 >=0;
         p2 : x_2 >=0;
cvx_end

f = [30 ; 20];
b = [24;36;4;-5];
A = [8 12 ; 12 12 ; 2 1 ; -1 -1];
cvx_begin
    variable x(2)
    dual variables a p;
    minimize( f'*x )
    subject to
         a : A*x >= b
         p : x >= 0;
cvx_end
```

Programmation linéaire forme standard

Matlab Optimization toolbox

```
X = LINPROG(f,A,b) attempts
to solve the linear programming problem:

min f'*x    subject to:   A*x <= b
  x

X = LINPROG(f,A,b,Aeq,beq) solves
the problem above while additionally
satisfying the equality constraints
Aeq*x = beq.

X = LINPROG(f,A,b,Aeq,beq,LB,UB)
defines a set of lower and upper
bounds on the design variables, X
...

f = [30 ; 20];
b = [24;36;4;-5];
A = [8 12 ; 12 12 ; 2 1 ; -1 -1];
[x c e o p] = linprog(f,-A,-b,[],[],0*f);
```

CVX: Matlab Software for Disciplined Convex Programming

Version 2.0 (beta), November 2012, Build 883

cvxr.com/cvx/

```
cvx_begin
    variable x(n)
    dual variables a p;
    minimize( f'*x  )
    subject to
        a : A*x == b;
        p : x >= 0;
cvx_end
```

Progresses in LP

in 1989

LP is a powerful modeling tool, "They are, however, theoretically complicated and computationally cumbersome"

from 1995 to 2015

	improvement factor
machine improvement	$\times 2^{10} = 1000 - 1600$
solver improvement	$\times 1000 - 3600$
formulation improvement	???
global improvement	$\times 1 - 5\ 10^6$

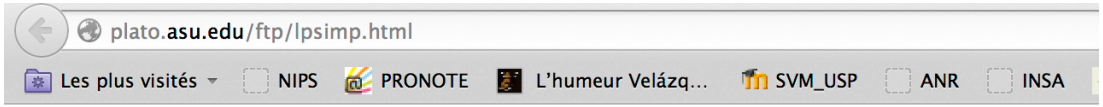
a year to solve $10 - 20$ years ago $\longrightarrow$ now 30 seconds

" linear techniques are nowadays mature, that is fast, robust, and are able to solve problems with up to millions of variables"

Programmation linéaire Solveurs

Software package	Matlab function
Open source	
GLPK	glpk for linear programming
COIN OR	(not matlab yet)
Commercial	
CVX	for convex optimization
Matlab	linprog for linear programming
CPLEX	cplexlp for linear programming
	cplexmiqp for mixed integer quadratic programming
	cplexmiqcp for mixed integer quadratically constrained pg
GUROBI	gurobi for LP, QP and MIP
Xpress	xpress for LP, QP and MIP
SCIP	opti_scip for LP, QP and MIP
MOSEK	...

Programmation linéaire Solveurs



23 Sep 2015 ==========
Benchmark of Simplex LP solvers
==========
H. Mittelmann (mittelmann@asu.edu)

Logfiles of these runs at: plato.asu.edu/ftp/lp_logs/

This benchmark was run on a Linux-PC (i7-4790K, 4.0GHz, 32GB).
The MPS-datafiles for all testcases are in one of (see column "s")

miplib.zib.de/ [1]
plato.asu.edu/ftp/lptestset/ [2]
www.netlib.org/lp/data/ [3,7]
www.sztaki.hu/~meszaros/public_ftp/lptestset/
(MISC[4], PROBLEMATIC[5], STOCHLP[6], INFEAS[8])

NOTE: files in [2-8] need to be expanded with emps in same directory!

The simplex methods were tested of the codes:

CPLEX-12.6.2	CPLEX
GUROBI-6.0.4	www.gurobi.com/
MOSEK-7.1.0.33	www.mosek.com
XPRESS-7.9.0	XPRESS (runs on 8 threads)
CLP-1.16.8	projects.coin-or.org/Clp (with openblas)
Google-GLOP	LP with Glop
SOPLEX-2.2.0	soplex.zib.de/
LP_SOLVE-5.5.2	lpsolve.sourceforge.net/
GLPK-4.55	www.gnu.org/software/glpk/glpk.html
MATLAB-R2015a	mathworks.com (dual-simplex)

Scaled shifted (by 10 sec) geometric mean of runtimes

	2.10	1.15	2.55	1	1.04	7.83	10.5	122	49.6	9.86
problem	CPXS	GRBS	MSKS	XPRS	CLP$	GLOP	SOPLX	LPSLV	GLPK	MATL
L1_sixm	856	119	671	103	525	f	11469	11965	5857	t
Linf_520c	t	8087	2021	267	22	272	t	523	1249	t
buildingen	7	6	306	23	170	313	531	14128	551	222
cont1	178	173	701	100	259	666	1578	398	f	f
cont11	6543	1117	1172	950	885	4905	4951	10537	f	943
cont4	177	170	689	118	240	351	1044	503	f	t
dano3mip	5	3	10	4	5	3	18	17455	4	18
dbic1	14	19	16	16	30	18	80	345	92	144
ds-big	245	354	363	255	175	331	533	t	1744	349
fome12	24	26	49	19	27	67	76	506	508	43
fome13	71	55	117	54	54	236	214	6498	2189	131
gen4	1	1	1	1	2	8	7	463	27	1
ken-18	1	1	4	1	2	49	77	1215	383	69

Programmation linéaire : formulation Standard

PL : forme standard

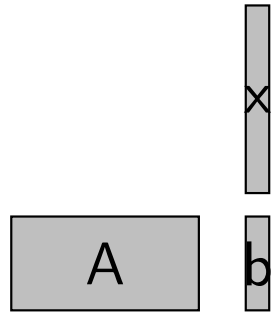
$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^m} & \mathbf{c}^\top \mathbf{z} \\ \text{avec} & A\mathbf{z} = \mathbf{b} \\ \text{et} & \mathbf{z} \geq 0 \end{cases}$$

- cout linéaire,
- contraintes linéaires
- $A \in \mathcal{M}(p, m)$

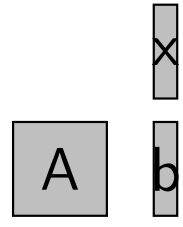
- m **variables (inconnues)** $\mathbf{z} = (z_1, z_2, \ldots, z_m)^\top$
- vecteur des couts (connu) $\mathbf{c} = (c_1, c_2, \ldots, c_m)^\top$
- la fonction objectif (de perte) $f(\mathbf{z}) = \mathbf{c}^\top \mathbf{z}$
- p **contraintes** : $p < m$ (moins de contraintes que d'inconnues)
- A une matrice p lignes (contraintes) et m colonnes (inconnues)
- $\mathbf{b} = (b_1, b_2, \ldots, b_p)^\top$ vecteur des *«second membres »*

Programmation linéaire : la nature du problème

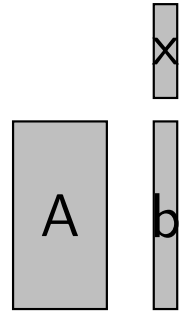
$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^m} & \mathbf{c}^\top \mathbf{z} \\ \text{avec} & A\mathbf{z} = \mathbf{b} \\ \text{et} & \mathbf{z} \geq 0 \end{cases}$$



Sous déterminé : il y a plus d'inconnues que d'équations



il y a autant d'équtions que d'inconnues



sur déterminé : il y a plus d'équations que d'inconnues

$A \in \mathcal{M}(p, m)$

Dans le cas qui nous intéresse le système est sous déterminé et la matrice A à plus de colonnes que de lignes : $p < m$

PL : réduction à la forme Standard

forme standard : cout linéaire, contraintes linéaires

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^m} & \mathbf{c}^\top \mathbf{z} \\ \text{avec} & A\mathbf{z} = \mathbf{b} \\ \text{et} & \mathbf{z} \geq 0 \end{cases} \qquad A \in \mathcal{M}(p, m)$$

un exemple simple $m = 2, p = 1$

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^2} & az_1 + bz_2 \\ \text{avec} & z_1 + z_2 \leq 1 \\ \text{et} & z_1 \geq 0, z_2 \geq 0 \end{cases}$$

il y a trois solutions selon les valeurs de (a, b).

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^2} & az_1 + bz_2 + 0z_3 \\ \text{avec} & z_1 + z_2 + z_3 = 1 \\ \text{et} & z_1 \geq 0, z_2 \geq 0, z_3 \geq 0 \end{cases}$$

PL : réduction à la forme Standard

forme standard : cout linéaire, contraintes linéaires

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^m} & \mathbf{c}^\top \mathbf{z} \\ \text{avec} & A\mathbf{z} = \mathbf{b} \\ \text{et} & \mathbf{z} \geq 0 \end{cases} \qquad A \in \mathcal{M}(p, m)$$

un exemple simple $m = 2, p = 1$

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^2} & az_1 + bz_2 \\ \text{avec} & z_1 + z_2 \leq 1 \\ \text{et} & z_1 \geq 0, z_2 \geq 0 \end{cases}$$

il y a trois solutions selon les valeurs de (a, b).

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^2} & az_1 + bz_2 + 0z_3 \\ \text{avec} & z_1 + z_2 + z_3 = 1 \\ \text{et} & z_1 \geq 0, z_2 \geq 0, z_3 \geq 0 \end{cases}$$

Domaine réalisable

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^m} & \mathbf{c}^\top \mathbf{z} \\ \text{avec} & A\mathbf{z} = \mathbf{b} \quad A \in \mathcal{M}(p,m) \\ \text{et} & \mathbf{z} \geq 0 \end{cases}$$

Domaine réalisable

C'est l'ensemble des vecteurs vérifiant les contraintes

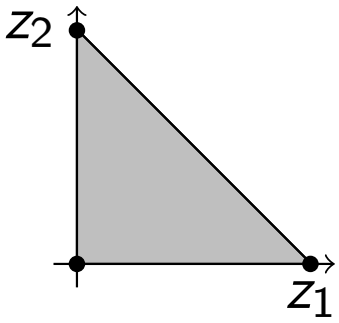
$$\mathcal{S}_a = \{\mathbf{z} \in \mathbb{R}^m | A\mathbf{z} = \mathbf{b};\ \mathbf{z} \geq 0\ \}$$

C'est un polytope (polyèdre) **convexe**

exemple :

$z_1 + z_2 \leq 1$ $\quad$ $z_1 \geq 0, z_2 \geq 0$

$z_1 + z_2 + z_3 = 1$ $\quad$ $z_1 \geq 0, z_2 \geq 0, z_3 \geq 0$



Formulation standard : des inégalités aux égalités

Introduction de nouvelles variables : des variables d'écart $\mathbf{e} \geq 0$

$$\mathbf{e} = (e_1, e_2, e_3, e_4)^\top$$

$$\begin{array}{rcl} 8x_1 + 12x_2 & \geq & 24 \\ 12x_1 + 12x_2 & \geq & 36 \\ 2x_1 + x_2 & \geq & 4 \\ x_1 + x_2 & \leq & 5 \\ x_1 & \geq & 0 \\ x_2 & \geq & 0 \end{array} \quad \Longrightarrow \quad \begin{array}{rcl} 8x_1 + 12x_2 - e_1 & = & 24 \\ 12x_1 + 12x_2 - e_2 & = & 36 \\ 2x_1 + x_2 - e_3 & = & 4 \\ x_1 + x_2 + e_4 & = & 5 \\ x_1 \geq 0, e_1 \geq 0, e_2 & \geq & 0 \\ x_2 \geq 0, e_3 \geq 0, e_4 & \geq & 0 \end{array}$$

$$\left\{ \begin{array}{ll} \min\limits_{\mathbf{z} \in \mathbb{R}^m} & \mathbf{d}^\top \mathbf{z} \\ \text{avec} & B\mathbf{z} \leq \mathbf{b} \\ \text{et} & \mathbf{z} \geq 0 \end{array} \right. \Longrightarrow \left\{ \begin{array}{ll} \min\limits_{\mathbf{z} \in \mathbb{R}^m, \mathbf{e} \in \mathbb{R}^p} & \mathbf{d}^\top \mathbf{z} \\ \text{avec} & B\mathbf{z} + \mathbf{e} = \mathbf{b} \\ \text{et} & \mathbf{z}, \mathbf{e} \geq 0 \end{array} \right. \Longrightarrow \left\{ \begin{array}{ll} \min\limits_{\mathbf{x} \in \mathbb{R}^{m+p}} & \mathbf{c}^\top \mathbf{x} \\ \text{avec} & A\mathbf{x} = \mathbf{b} \\ \text{et} & \mathbf{x} \geq 0 \end{array} \right.$$

$$\mathbf{x} = (\mathbf{z}; \mathbf{e}), \qquad \mathbf{c} = (\mathbf{d}; 0); \qquad A = \begin{array}{|c|c|} \hline B & I \\ \hline \end{array}$$

PL : réduction à la forme Standard

La cas des variables non positives

$$\begin{cases} \min\limits_{x,y,z} & x+3y+4z \\ \text{avec} & x+2y+z=5 \\ & 2x+3y+z=6 \\ \text{et} & y \geq 0, z \geq 0 \end{cases}$$

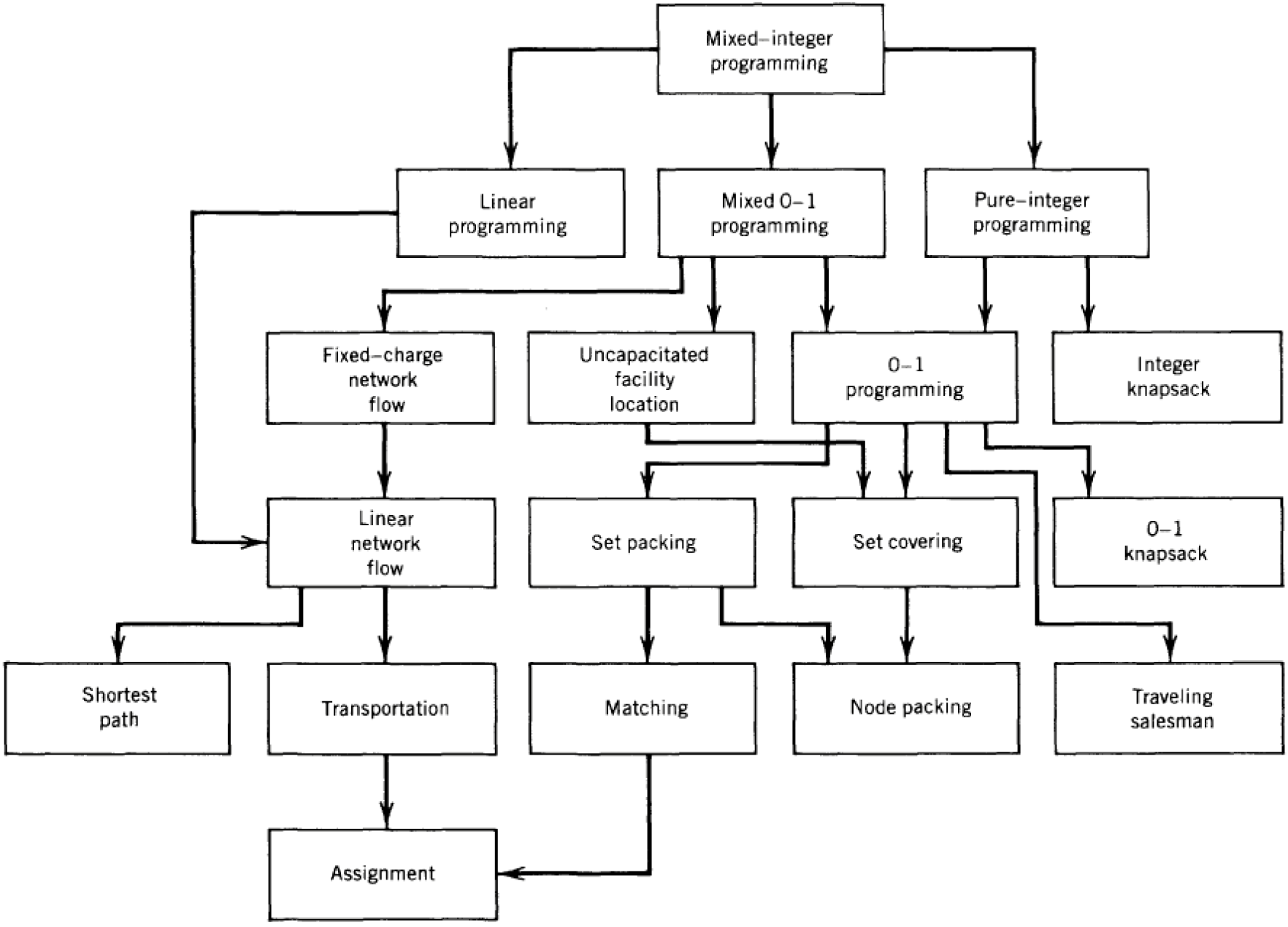
Trois possibilités :

- on ne fait rien...
- on remplace x par la différence de deux variables contraintes : $x = x_p - x_m$ avec $x_p \geq 0$ et $x_m \geq 0$

$$\begin{cases} \min\limits_{x_p,x_m,y,z} & x_p - x_m + 3y + 4z \\ \text{avec} & x_p - x_m + 2y + z = 5 \\ & 2x_p - 2x_m + 3y + z = 6 \\ \text{et} & x_p \geq 0, x_m \geq 0, y \geq 0, z \geq 0 \end{cases}$$

- on élimine x à l'aide d'une contrainte

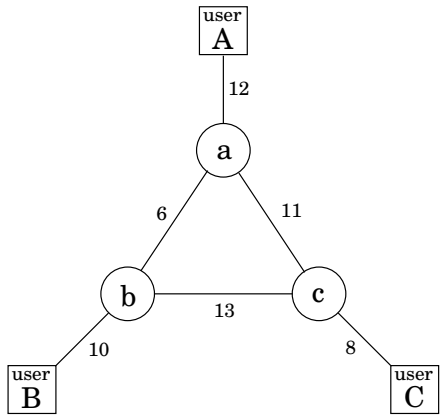
Classification des programmes linéaires



Un exemple de programmation linéaire

Maximisation d'un flot :

- prix : A-B 3 €, B-C 2€ et A-C 4€
- par contrat, chaque connexion doit au moins recevoir 2 unités
- trouver comment maximiser les revenus sur le réseau

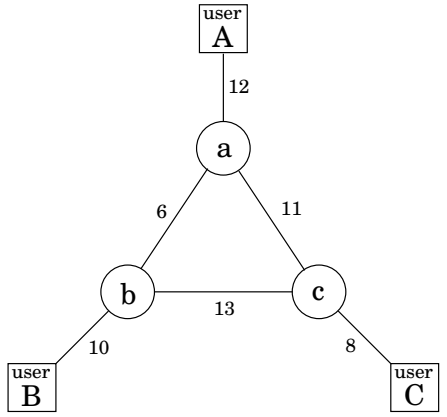


voir Algorithms, S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, 2006, page 195

Un exemple de programmation linéaire

Maximisation d'un flot :

- prix : A-B 3 €, B-C 2€ et A-C 4€
- par contrat, chaque connexion doit au moins recevoir 2 unités
- trouver comment maximiser les revenus sur le réseau



On introduit les variables x_i : x_1 (AB direct),x_2 (AB long), x_3 (BC direct), x_4 (BC long), x_5 (AC direct), x_6 (AC long)

$$\begin{cases} \max\limits_{\mathbf{x}\in\mathbb{R}^6} & 3x_1 + 3x_2 + 2x_3 + 2x_4 + 4x_5 + 4x_6 \\ \text{avec} & x_1 + x_2 + x_3 + x_4 \leq 10 \qquad \text{arrête B-b} \\ & x_1 + x_4 + x_6 \leq 6 \qquad \text{arrête a-b} \\ & x_1 + x_4 + x_6 \leq 6 \qquad \text{arrête a-b} \\ & x_1 + x_2 \geq 2 \qquad \text{traffic minimum} \\ & \dots \\ \text{et} & x_i \geq 0 \end{cases}$$

voir Algorithms, S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, 2006, page 195

Un exemple de programmation linéaire

Le problème de transport :

- d ports de départ avec chacun une quantité $a_i, i = 1, d$ à envoyer
- r ports d'arrivée avec chacun une quantité $b_j, j = 1, r$ à recevoir
- c_{ij} le cout de transport d'un port a_i vers un port b_j est connu
- trouver comment affréter les bateaux pour un cout minimal

Un exemple de programmation linéaire

Le problème de transport :

- d ports de départ avec chacun une quantité $a_i, i = 1, d$ à envoyer
- r ports d'arrivée avec chacun une quantité $b_j, j = 1, r$ à recevoir
- c_{ij} le cout de transport d'un port a_i vers un port b_j est connu
- trouver comment affréter les bateaux pour un cout minimal

On introduit les variables x_{ij} : la quantité de produit transporté de i vers j.

$$\begin{cases} \min\limits_{\mathbf{z}\in\mathbb{R}^{d\times r}} & \sum\limits_{i=1}^{d}\sum\limits_{j=1}^{r} c_{ij}x_{ij} \\ \text{avec} & \sum\limits_{j} x_{ij} = a_i; \quad i = 1, d \\ & \sum\limits_{i} x_{ij} = b_j; \quad j = 1, r \\ \text{et} & x_{ij} \geq 0 \end{cases}$$

Conclusion

- Optimisation sous contrainte : programmation linéaire
  - flots
  - transport
  - plus court chemin
- L'importance de la forme standard
  - réduction des problèmes
- Il existe maintenant d'autres algorithmes plus puissants : points intérieurs
- le Lièvre L2 et la tortue L1

Example : a vous

L'entreprise Cannes United dispose de 3 usines de fabrication (A,B et C). Il souhaite minimiser ses couts de transports entre les sites de fabrication et ses 4 magasins (W,X,Y et Z).

Usine	Production
A	400
B	1500
C	900
Total	2800

Magasins	Demande
W	700
X	600
Y	1000
Z	500
Total	2800

Nous avons aussi la matrice des couts de transports :

	W	X	Y	Z
A	20	40	70	50
B	100	60	90	80
C	10	110	30	200

Par exemple, le cout de transport vers Y d'une canne fabriquée à A est de 30 euros.

1. formuler le problème comme un programme linéaire,
2. donner la solution du problème
3. donner la formulation duale