Chapitre 6 : Problèmes d'affectations

J.-F. Scheid

2011–2012

Plan du chapitre

I. Affectation simple

1. Introduction
2. Modélisation par un PL en variables binaires.
3. Modélisation par flot maximal.
4. Résolution par l'algorithme de Ford-Fulkerson.

II. Affectation multiple

1. Introduction
2. Modélisations et résolution (Ford-Fulkerson).

I. Affectation simple
1) Introduction

Un exemple : 4 tâches (demandes) $C_1, \cdots, C_4$ doivent être réalisées en disposant de 3 machines (offres) $L_1, \cdots, L_3$. Chaque machine ne peut effectuer que certaines tâches bien précises.

Les tâches permises et non-permises pour chaque machine, sont indiquées dans **le tableau des cases admissibles** :

	C_1	C_2	C_3	C_4
L_1				
L_2				
L_3				

: affectation non-permise

Chaque tâche ne doit pas être effectuée par plus d'une machine et chaque machine ne peut pas effectuer plus d'une tâche.

Affectation simple :

- Chaque demande (tâche) ne peut pas être traitée par plus d'une offre (machine).
- Chaque offre (machine) ne peut pas traiter plus d'une demande (tâche).

Remarque. Une demande peut ne pas être traitée du tout et une offre peut n'être affectée à aucune demande.

Problème d'affectation

Trouver le maximum d'affectations possibles.

2) Modélisation par un PL en variables binaires

On introduit les variables t_{ij} qui indiquent si l'offre L_i est affectée à la demande C_j :

$$t_{ij} = \begin{cases} 1 & \text{si } L_i \text{ est affectée à } C_j \\ 0 & \text{sinon} \end{cases}$$

pour $(i, j) \in \mathcal{U}$ ensemble des indices **admissibles**

Exemple.

	C_1	C_2	C_3	C_4
L_1				
L_2				
L_3				

$$\mathcal{U} = \{ \ (1,1), \\ (2,1), (2,2), (2,3), (2,4), \\ (3,1), (3,2) \ \}$$

2) Modélisation par un PL en variables binaires

Problème de programmation linéaire (primal)

$$
(P_1) \begin{cases}
\max\limits_{t_{ij}} \left[F_1 = \sum\limits_{(i,j)\in\mathcal{U}} t_{ij} \right] & \leftarrow \quad \text{maximisation du nb d'affectations} \\
\forall i, \ \sum\limits_{j} t_{ij} \leq 1 & \leftarrow \quad \text{offre } L_i \text{ affectée à une demande au plus} \\
\forall j, \ \sum\limits_{i} t_{ij} \leq 1 & \leftarrow \quad \text{demande } C_j \text{ affectée à une offre au plus} \\
\forall (i,j) \in \mathcal{U}, \ \ t_{ij} \geq 0 &
\end{cases}
$$

Exemple.

	C_1	C_2	C_3	C_4
L_1				
L_2				
L_3				

$$\max_{\mathbf{t}} F(\mathbf{t}) = \mathbf{c}^\top \mathbf{t}$$

$$\begin{cases} A\mathbf{t} \leq \mathbf{b} \\ \mathbf{t} \geq 0 \end{cases}$$

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}; \quad \mathbf{t} = \begin{pmatrix} t_{11} \\ t_{21} \\ t_{22} \\ t_{23} \\ t_{24} \\ t_{31} \\ t_{32} \end{pmatrix}; \quad \mathbf{b} = \mathbf{c} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix};$$

Propriété

Toute solution de base optimale t_{ij}^* de (P_1) vérifie $t_{ij}^* \in \{0, 1\}$.

En effet,

- on a $0 \leq t_{ij}^* \leq 1$
- on peut montrer que la solution de base optimale est **entière** (cf. chapitre 4, PL en variables entières)

Le problème primal (P_1) s'interprète de la façon suivante :

On veut trouver un maximum de cases admissibles 2 à 2 *indépendantes* c'est-à-dire *ni sur la même ligne, ni sur la même colonne*.

Par exemple,

	C_1	C_2	C_3	C_4
L_1	1			
L_2	0	1	0	0
L_3	0	0		

Le problème primal (P_1) admet le dual suivant.

Dual

$$(D_1) \quad \begin{cases} \min\limits_{l_i, k_j} \left[G_1 = \sum\limits_i l_i + \sum\limits_j k_j \right] \\ \forall (i,j) \in \mathcal{U}, \; l_i + k_j \geq 1 \\ \forall i, \; l_i \geq 0 \\ \forall j, \; k_j \geq 0 \end{cases}$$

Remarque. On montre qu'à l'optimum, les variables duales sont entières et l_i, $k_j \in \{0, 1\}$.

Définition

Un **support** est un ensemble de lignes et de colonnes qui couvrent toutes les cases admissibles du tableau.

	C_1	C_2	C_3	C_4
L_1				
L_2				
L_3				

Supports : $\{L_1, L_2, L_3\}$, $\{C_1, C_2, C_3, C_4\}$, $\{C_1, L_2, L_3\}$, $\cdots$

Remarque. Si $l_i = 1$ alors L_i est dans le support. De même si $k_j = 1$ alors C_j est dans le support.

$\rightarrow$ Le problème dual correspond à la recherche d'un **support minimal** (i.e. support de cardinal minimal)

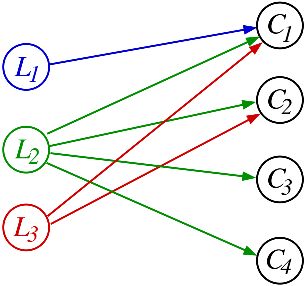
3) Modélisation par flot maximal

Au tableau des cases admissibles, on peut associer un diagramme sagital avec un graphe biparti.

Définition

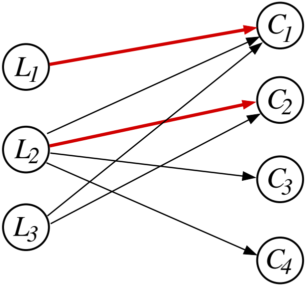
Un graphe est dit **biparti** si l'ensemble de ses sommets peut être partitionné en 2 sous-ensembles X et Y tels que toute arête possède une extrémité dans X et l'autre dans Y.

	C_1	C_2	C_3	C_4
L_1				
L_2				
L_3				



Remarque. Sur le graphe biparti associé au tableau des cases admissibles, le problème d'affectation modélisé par le problème primal (P_1), correspond à rechercher le maximum d'arêtes 2 à 2 *non-adjacentes* c'est-à-dire qui n'ont *ni la même origine, ni le même sommet terminal*.

	C_1	C_2	C_3	C_4
L_1	1			
L_2	0	1	0	0
L_3	0	0		

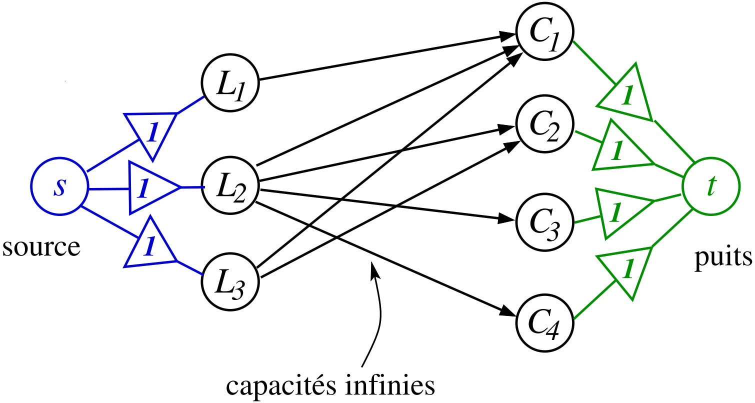


Graphe biparti complété.

Soit G le graphe biparti associé au tableau des cases admissibles avec X et Y les deux sous-ensembles tels que toute arête a une extrémité dans X et l'autre dans Y.

- On ajoute une **source** $s \in X$ et des arêtes (s, i) de capacité $c(s, i) = 1$ pour tous les sommets i de X.
- On ajoute un **puits** $t \in Y$ et des arêtes (j, t) de capacité $c(j, t) = 1$ pour tous les sommets j de Y.
- A chaque arête (i, j) du graphe G initial, on associe une capacité **infinie**

On obtient ainsi un **graphe biparti complété**



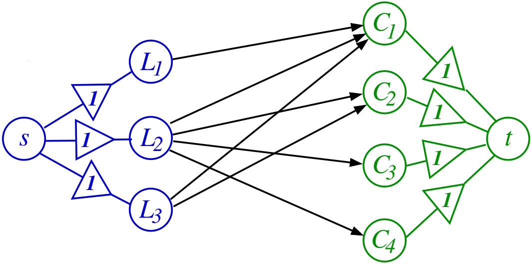
Le problème d'affectation est un problème de flot maximal à travers le graphe biparti complété.

Flot maximal : problème primal (P_2)

$$\max\left[F_2 = v\right]$$

$$(P_2)\quad \left\{\begin{array}{lr} \left\{\begin{array}{ll} (s) & -v + \sum_i f_{si} = 0 \\ \forall L_i, & -f_{si} + \sum_j f_{ij} = 0 \\ \forall C_j, & -\sum_i f_{ij} + f_{jt} = 0 \\ (t) & -\sum_j f_{jt} + v = 0 \end{array}\right. \\ \left\{\begin{array}{ll} \forall L_i, & f_{si} \leq 1 \\ \forall C_j, & f_{jt} \leq 1 \\ \forall (i,j), & f_{ij} \geq 0 \\ & v \text{ de signe quelconque} \end{array}\right. \end{array}\right.$$

Exemple.



$$\max_{\mathbf{f},v} [F_2 = v] \quad \begin{cases} A\mathbf{f} + \mathbf{v} = 0 \\ \mathbf{f} \leq 1 \\ \mathbf{f} \geq 0 \end{cases}$$

$\mathbf{f} = \left(\; f_{s1}, f_{s2}, f_{s3} \,|\, f_{11}, f_{21}, f_{22}, f_{23}, f_{24}, f_{31}, f_{32} \,|\, f_{1t}, f_{2t}, f_{3t}, f_{4t} \; \right)^{\top} \in \mathbb{R}^{14}$

$\mathbf{v} = \left(\; -v, 0, 0, 0, 0, 0, 0, 0, +v \; \right)^{\top} \in \mathbb{R}^{9}$;

Matrice $A \in \mathcal{M}_{9\times 14}$:

$$A = \left(\begin{array}{ccc|ccccccc|cccc}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1
\end{array}\right)$$

Propriétés

1. Toute solution de base optimale f_{ij}^* de (P_2) vérifie $f_{ij}^* \in \{0, 1\}$: à l'optimum, les flots des arêtes valent 0 ou 1 :
2. Les 2 problèmes primaux (P_1) et (P_2) sont équivalents.

Preuve :

1) toute solution de base optimale de (P_2) est **entière** (cf. Chapitre 4, PL en nb entiers).

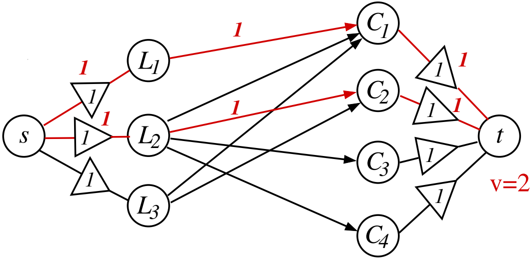
2) prendre $t_{ij} = f_{ij}$ (exercice)

4) Résolution par Ford-Fulkerson

Algorithme de Ford-Fulkerson pour calculer le flot maximal à travers le graphe biparti complété.

1. Initialisation du flot (coin nord-ouest).
   On attribue les affectations en partant de la 1ère ligne et en allant de gauche à droite.

	C_1	C_2	C_3	C_4
L_1	1			
L_2	0	1	0	0
L_3	0	0		



② Amélioration du flot par l'algorithme de Ford-Fulkerson

⋆ *marquage pile largeur* (par ex.) : à partir du haut de la pile, on marque et on empile tous les sommets successeurs non encore marqués.

⋆ inutile d'indiquer le tableau des améliorations ε : on a toujours $\varepsilon = 1$

$\mathbb{E}$	**s**	$\mathbf{L_3}$	$C_1, \mathbf{C_2}$	$\mathbf{L_2}$	$C_3, \mathbf{C_4}$	**t**
`orig`	—	s	L_3	$-C_2$	L_2	C_4

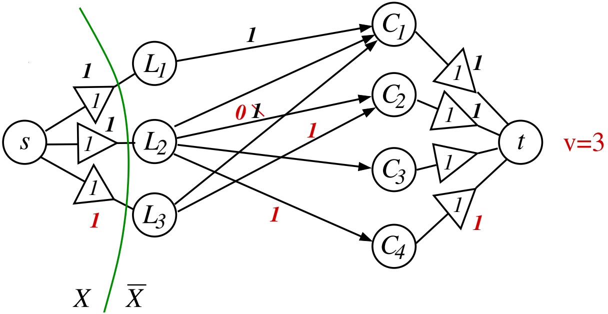


Tableau correspondant

	C_1	C_2	C_3	C_4
L_1	1			
L_2	0	0	0	1
L_3	0	1		

- Sur le graphe : $(X, \overline{X})$ coupe minimale $\Rightarrow$ flot maximal $\max v = 3$.
- Dans le tableau : nombre d'affectations $=$ min(nb de ligne, nb de colonne) $= 3 \Rightarrow$ nombre maximal d'affectation$=3$

II. Affectation multiple
1) Introduction

On reprend l'exemple précédent (affectation simple) mais cette fois chaque machine (offre) peut réaliser plusieurs tâches (demande) :

- la machine L_i peut être utilisée au plus a_i fois ($a_i \in \mathbb{N}^*$).

On suppose aussi que chaque tâche peut utiliser un certain nombre de fois les différentes machines :

- la tâche C_j peut être réalisée en utilisant au plus b_j machines ($b_j \in \mathbb{N}^*$).

Les offres et demandes sont indiquées dans le tableau des cases admissibles :

	C_1	C_2	C_3	C_4	a_i
L_1					6
L_2					7
L_3					3
b_j	10	3	4	6	

Dans l'exemple ci-dessus, la machine L_1 peut être utilisée au plus 6 fois. La tâche C_1 peut être effectuée par au plus 10 machines...

$\rightarrow$ Trouver le maximum d'affectation possible

2) Modélisations

a) Programmation linéaire

Variable t_{ij} = nombre de fois que l'offre L_i est utilisée par la demande C_j
= nombre d'affectations de l'offre L_i à la demande C_j

$$\begin{cases} \max\limits_{t_{ij}} \left[F = \sum\limits_{(i,j)\in\mathcal{U}} t_{ij} \right] \\ \forall i, \ \sum\limits_j t_{ij} \leq a_i \quad \text{(offre)} \\ \forall j, \ \sum\limits_i t_{ij} \leq b_j \quad \text{(demande)} \\ \forall (i,j) \in \mathcal{U}, \ \ t_{ij} \geq 0 \end{cases}$$

Remarque. Si les a_i et b_j sont entiers alors à l'optimum les t_{ij} sont entiers.

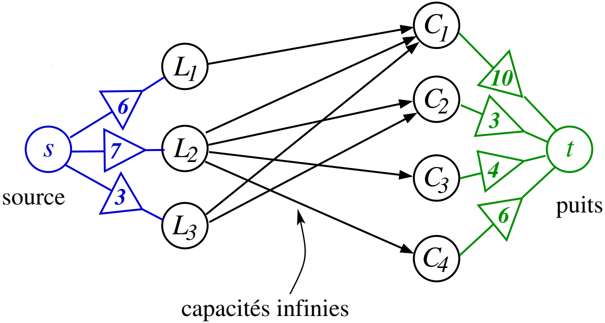
b) Modélisation par flot maximal

Le problème de l'affectation multiple peut se modéliser par la recherche d'un flot maximal à travers un graphe **biparti complété** :

- On ajoute une **source** s et des arêtes (s, i) reliant s aux offres L_i avec des capacités $c(s, i) = a_i$.
- On ajoute un **puits** t et des arêtes (j, t) reliant les demandes C_j à t avec des capacités $c(j, t) = b_i$.
- Pour chaque arête (i, j) entre L_i et C_j, on considère une capacité **infinie**.

Exemple

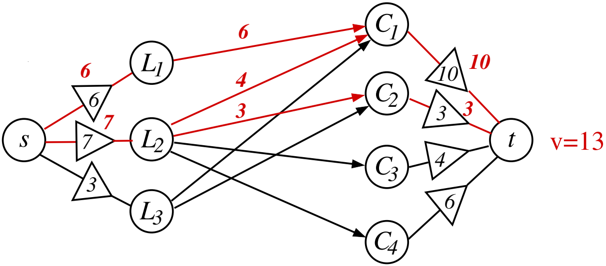
	C_1	C_2	C_3	C_4	a_i
L_1					6
L_2					7
L_3					3
b_j	10	3	4	6	



Algorithme de Ford-Fulkerson pour calculer le flot maximal à travers le graphe biparti complété.

1. Initialisation du flot (coin nord-ouest).

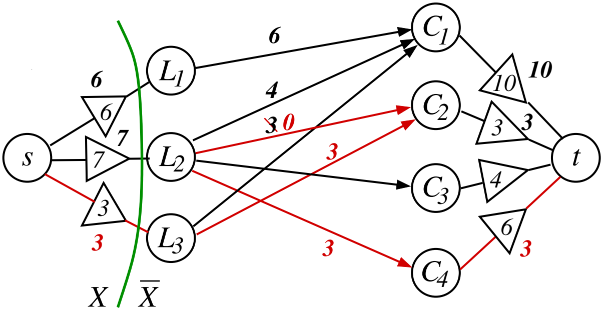
	C_1	C_2	C_3	C_4	a_i
L_1	6				6
L_2	4	3	0	0	7
L_3	0	0			3
b_j	10	3	4	6	



2. Amélioration du flot par Ford-Fulkerson

Marquage pile largeur (par ex.)

$\mathbb{E}$	**s**	$\mathbf{L_3}$	$C_1, \mathbf{C_2}$	$\mathbf{L_2}$	$C_3, \mathbf{C_4}$	**t**
orig	$-$	s	L_3	$-C_2$	L_2	C_4
ε	∞	3	3	3	3	$\varepsilon = 3$



v=16

Tableau correspondant

	C_1	C_2	C_3	C_4	a_i
L_1	6				6
L_2	4	0	0	3	7
L_3	0	3			3
b_j	10	3	4	6	

- Sur le graphe : $(X, \overline{X})$ coupe minimale $\Rightarrow$ flot maximal $\max v = 16$.
- Dans le tableau : il n'y a plus d'affectation possible selon les lignes $\Rightarrow$ nombre maximal d'affectation=16