Cours #4 : Modélisation par les flots

But : Modéliser des problèmes par des models de flot et modéliser les problèmes de flot comme programmes linéaires.

Réseau de transport : $N=(V,A,s,t,c)$ ou $G=(V,A)$ est un graphe, V - l'ensemble des sommets, A - l'ensemble d'arcs (arêtes), s - la source, t - le puits et $c: A \to \mathbb{R}^+$ les capacités.

Flot : $f: A \to \mathbb{R}^+$ telle que :

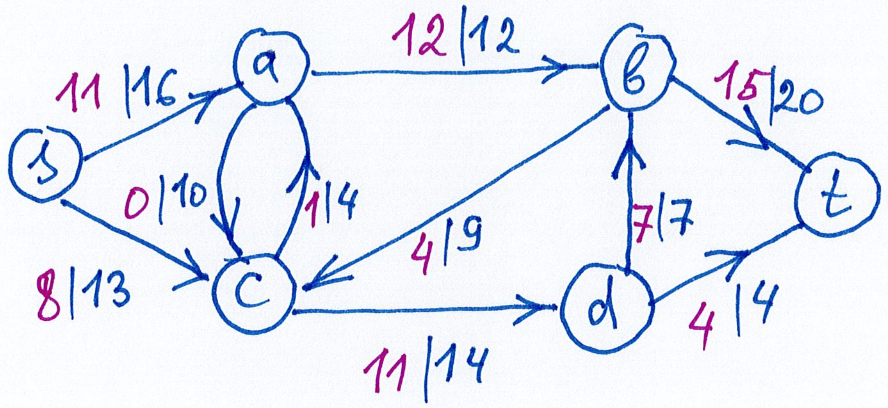
Contrainte de capacité : $f(\vec{uv}) \leq c(\vec{uv})$ $\forall \vec{uv} \in A$

Conservation du flot : $\sum_{\vec{uv} \in \Delta^+(v)} f(\vec{uv}) = \sum_{\vec{vu} \in \Delta^-(v)} f(\vec{vu})$ $\forall v \in V$, $v \neq s,t$

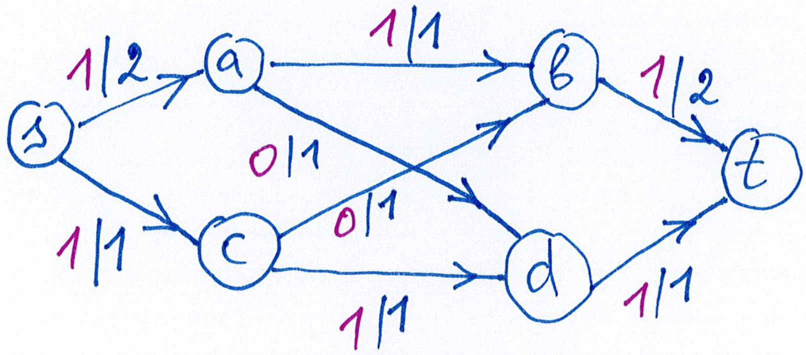
$\Delta^+(v)$ = les arcs entrants dans v

$\Delta^-(v)$ = les arcs sortants de v.

Exemple:



$val(f) = 19$



$val(f) = 2$

Exces d'un sommet: $exces(v) = \sum_{\vec{uv} \in \Delta^{+}(v)} f(\vec{uv}) - \sum_{\vec{vu} \in \Delta^{-}(v)} f(\vec{vu})$

le flot entrant dans v moins le flot sortant

Valeur d'un flot f: $val(f) = exces(t)$

Lemme 1: $val(f) = -exces(s)$.

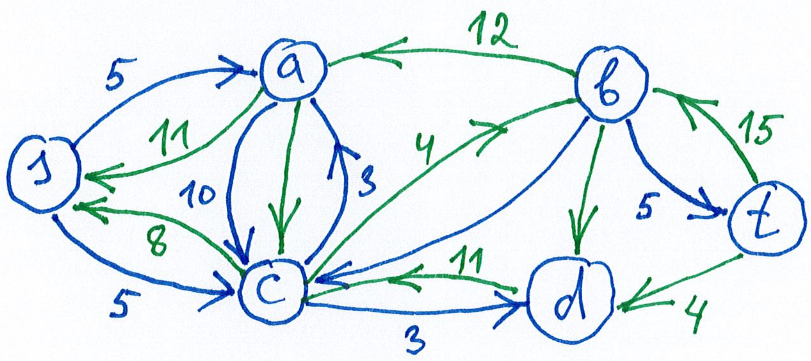
Flot maximum: flot de valeur maximum

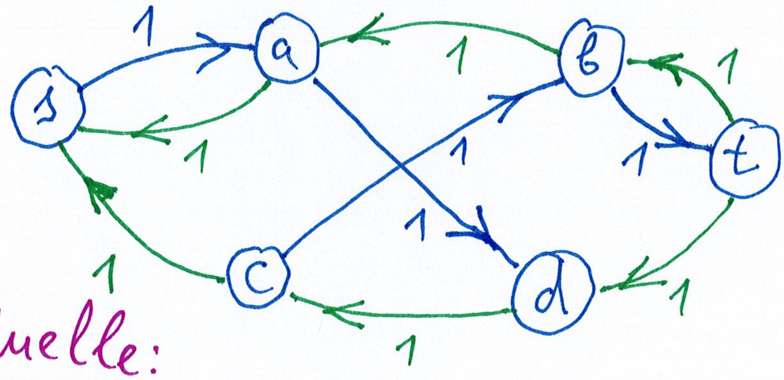
Rappels sur les flots max (déjà vu en L3)

(1) Réseau résiduel d'un flot f : N_f

- même ensemble des sommets que N
- $\overrightarrow{uv}$ est un arc de N_f ssi :
  - soit $\overrightarrow{uv}$ est un arc de N et $f(\overrightarrow{uv}) < c(\overrightarrow{uv})$ (arc non-saturés)
  - soit $\overrightarrow{vu}$ est un arc de N et $f(\overrightarrow{vu}) > 0$ (arc arrière)

Exemple:





Capacité résiduelle:

$c_f(\overrightarrow{uv}) = c(\overrightarrow{uv}) - f(\overrightarrow{uv})$ pour les arcs non-saturé

$c_f(\overrightarrow{uv}) = f(\overrightarrow{vu})$ pour les arcs arrière.

Chaine augmentante: tout chemin P de s à t dans N_f.

Soit $\delta(P) = \min\{c_f(\vec{uv}) : \vec{uv} \in P\} > 0$.

Augmentation du flot f: pour tout arc $\vec{uv}$ du réseau initial N, faire:

$$f'(\vec{uv}) = \begin{cases} f(\vec{uv}) + \delta(P) & \text{si } \vec{uv} \in P \text{ non-saturé} \\ f(\vec{uv}) - \delta(P) & \text{si } \vec{vu} \text{ arc arière de } P \\ f(\vec{uv}) & \text{sinon.} \end{cases}$$

Lemme 2: f' est un flot de valeur $val(f) + \delta(P)$.

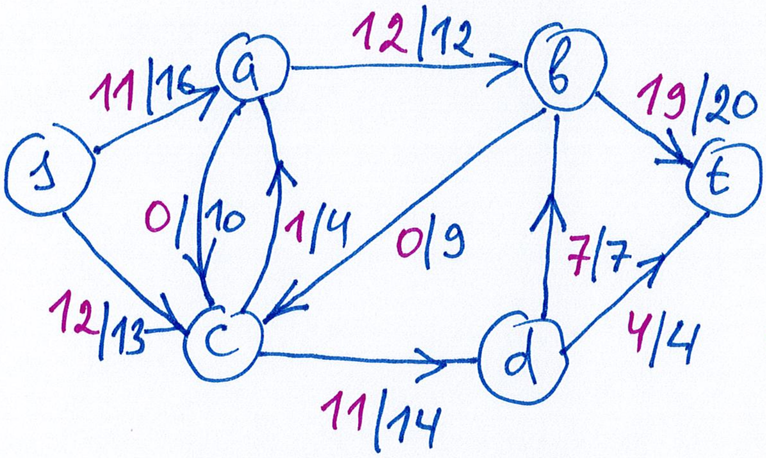
Algorithme Ford-Fulkerson

1. Partir d'un flot f quelconque (disons le flot nul)
2. Tant que possible augmenter le flot courrant.

Cet algorithme est pseudo-polynomial, se termine pas toujours mais se termine pour des capacites entiéres (ou rationnelles).

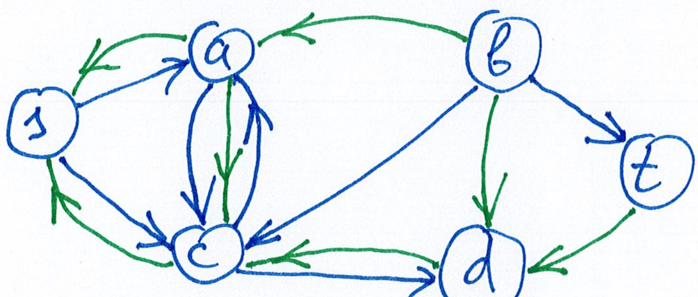
Exemple: (a) $s \to c \to b \to t$

(b) $s \to a \to d \to c \to b \to t$

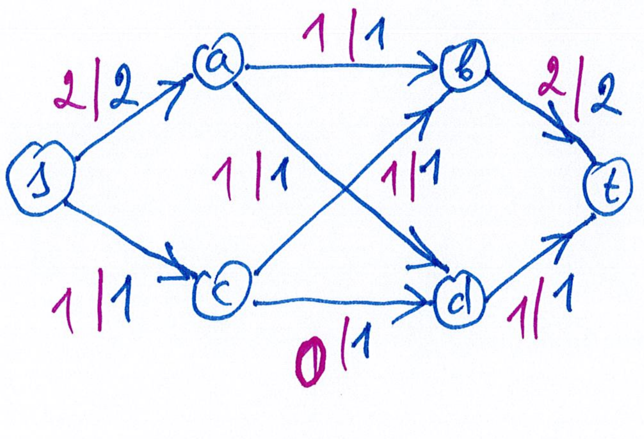


$val(f) = 23$

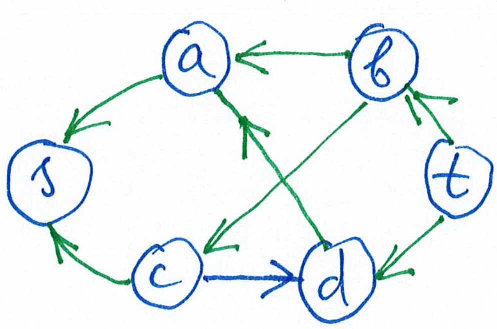
$\Downarrow$



N_f



$\Rightarrow$

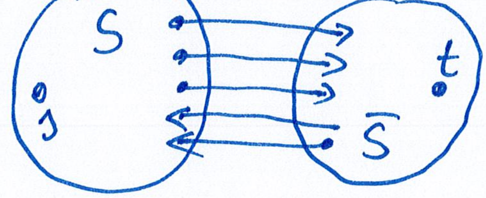


$val(f) = 3$

N_f

L'algo de Ford-Fulkerson devient fortement polynomial (avec une complexité $O(nm^2)$) si la chaine augmentante P chaque fois est un plus court chemin (en nombre d'arêtes) du réseau N_f.

(s,t)-coupe : une partition $S \uplus \bar{S}$ de V t.q. $s \in S$ et $t \in \bar{S}$.



Capacité $c(S,\bar{S}) = \sum_{\substack{\vec{uv} \in A \\ u \in S, v \in \bar{S}}} c(\vec{uv})$

Théorème 1 : (Dualité faible) Pour tout flot f et toute (s,t)-coupe $(S,\bar{S})$, $val(f) \leq c(S,\bar{S})$.

(Dualité forte) Il existe un flot f^* et une (s,t)-coupe $(S^*,\bar{S}^*)$ tels que $val(f^*) = c(S^*,\bar{S}^*)$.

Alors f^* est un $\boxed{\text{flot max}}$ et $(S^*,\bar{S}^*)$ une $\boxed{\text{coupe minimum}}$

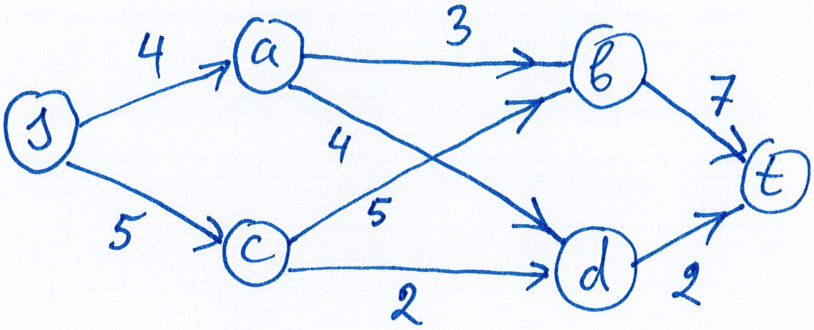
Rémarque: Le Théorème 1 est un cas particulier de la dualité de la prog. linéaire

Les flots max comme programmes linéaires:

- une variable $f_{\vec{uv}}$ pour chaque arc $\vec{uv}$ de N
- une inegalité $f_{\vec{uv}} \leq c_{\vec{uv}}$ pour chaque arc $\vec{uv}$
- une egalité $\displaystyle\sum_{\vec{uv} \in \Delta^+(v)} f_{\vec{uv}} - \sum_{\vec{vu} \in \Delta^-(v)} f_{\vec{vu}} = 0$
- maximiser l'exces de t $\displaystyle\sum_{\vec{ut} \in \Delta^+(t)} f_{\vec{ut}} - \sum_{\vec{tv} \in \Delta^-(t)} f_{\vec{tv}}$.

$$
\begin{cases}
\text{maximiser} \displaystyle\sum_{\vec{ut} \in \Delta^+(t)} f_{\vec{ut}} - \sum_{\vec{tv} \in \Delta^-(t)} f_{\vec{tv}} \\[2ex]
f_{\vec{uv}} \leq c_{\vec{uv}} \qquad \forall\, \vec{uv} \in A \\[2ex]
\displaystyle\sum_{\vec{uv} \in \Delta^+(v)} f_{\vec{uv}} - \sum_{\vec{vu} \in \Delta^-(v)} f_{\vec{vu}} = 0 \qquad \forall\, v \in V \setminus \{s,t\} \\[2ex]
f_{\vec{uv}} \geq 0 \qquad \forall\, \vec{uv} \in A
\end{cases}
$$

Exemple:



$$\begin{cases}
\max \quad f_{\vec{bt}} + f_{\vec{dt}} \quad (\text{où } f_{\vec{sa}} + f_{\vec{sc}}) \\
\left.\begin{array}{l} f_{\vec{sa}} \leq 4 \\ f_{\vec{sc}} \leq 5 \\ \vdots \\ \vdots \end{array}\right\} \text{8 inegalites} \\
f_{\vec{sa}} = f_{\vec{ab}} + f_{\vec{ad}} \\
f_{\vec{sc}} = f_{\vec{cb}} + f_{\vec{cd}} \\
f_{\vec{ab}} + f_{\vec{cb}} = f_{\vec{bt}} \\
f_{\vec{ad}} + f_{\vec{cd}} = f_{\vec{dt}} \\
f_{\vec{sa}}, \ldots\ldots, f_{\vec{dt}} \geq 0
\end{cases}$$

2. Circulation avec demandes: $N=(V,A,c,d)$

$c: A \to \mathbb{R}^+$ - la fonction des capacités

$d: V \to \mathbb{R}$ - la fonction des démandes

$d_v > 0$ - un puits qui requiert d_v unités de flot

$d_v < 0$ - une source qui produit $-d_v$ unités de flot

$d_v = 0$ - sommet classique

Une circulation avec demandes: $f: A \to \mathbb{R}^+$ telle que:

Loi de capacité: $f(\vec{uv}) \leq c(\vec{uv}) \quad \forall \vec{uv} \in A$

Loi de conservation: $\sum\limits_{\vec{uv} \in \Delta^+(v)} f(\vec{uv}) - \sum\limits_{\vec{vu} \in \Delta^-(v)} f(\vec{vu}) = d_v \quad \forall v \in V$

Lemme 3: Si il existe une circulation avec demandes, alors $\sum\limits_{v \in V} d_v = 0$, i.e.

$$\sum_{v: d_v > 0} d_v = -\sum_{v: d_v < 0} d_v =: D.$$

Rémarque: Le problème de flot de valeur d se reduit à la circulation en posant $d_s = -d$, $d_t = d$ et $d_v = 0 \quad \forall v \neq s, t$.

Rémarque: Le problème de circulation se réduit au flot max.

But: Décider si il existe $f : A \to \mathbb{R}^+$ t.q.

- $l(\vec{uv}) \leq f(\vec{uv}) \leq c(\vec{uv}) \quad \forall \vec{uv} \in A$
- $\displaystyle\sum_{\vec{uv} \in \Delta^+(v)} f(\vec{uv}) - \sum_{\vec{vu} \in \Delta^-(v)} f(\vec{vu}) = d_v \quad \forall v \in V$

Program linéaire:

$$\begin{cases} f_{\vec{uv}} \leq c_{\vec{uv}} & \forall \vec{uv} \in A \\ f_{\vec{uv}} \geq l_{\vec{uv}} & \forall \vec{uv} \in A \\ \displaystyle\sum_{\vec{uv} \in \Delta^+(v)} f_{\vec{uv}} - \sum_{\vec{vu} \in \Delta^-(v)} f_{\vec{vu}} = d_v & \forall v \in V \end{cases}$$

Flot:

$$\begin{cases} \text{maximiser} \displaystyle\sum_{\vec{vt} \in \Delta^+(t)} f_{\vec{vt}} - \sum_{\vec{tv} \in \Delta^-(t)} f_{\vec{tv}} \\ f_{\vec{uv}} \leq c_{\vec{uv}} & \forall \vec{uv} \in A \\ f_{\vec{uv}} \geq l_{\vec{uv}} & \forall \vec{uv} \in A \\ \displaystyle\sum_{\vec{uv} \in \Delta^+(v)} f_{\vec{uv}} - \sum_{\vec{vu} \in \Delta^-(v)} f_{\vec{vu}} = 0 & \forall v \in V \\ & v \neq s, t \end{cases}$$

Rémarque: Si $l(\vec{uv}) = c(\vec{uv})$ cela oblige de passer par l'arc $\vec{uv}$ en quantité $c(\vec{uv})$.

Circulation avec demandes est equivalent à la féasabilité du programme lineaire suivant:

$$\begin{cases} f_{\vec{uv}} \leq c_{\vec{uv}} & \forall \vec{uv} \in A \\[2ex] \displaystyle\sum_{\vec{uv} \in \Delta^+(v)} f_{\vec{uv}} - \sum_{\vec{vu} \in \Delta^-(v)} f_{\vec{vu}} = d_v & \forall v \in V \\[2ex] f_{\vec{uv}} \geq 0 & \forall \vec{uv} \in A \end{cases}$$

3. Circulations et flot avec demandes et bornes inf et sup de capacités:

Circulation : $N = (V, A, \ell, c, d)$

$\ell : A \to \mathbb{R}^+$ – les bornes inf des capacites

$c : A \to \mathbb{R}^+$ – les bornes sup (classiques) des capacites

$d : V \to \mathbb{R}$ – les demandes des sommets

~~On cherche $f : V \to \mathbb{R}^+$ t.q.~~

Flot de coût minimum : $N = (V, A, s, t, c, q)$

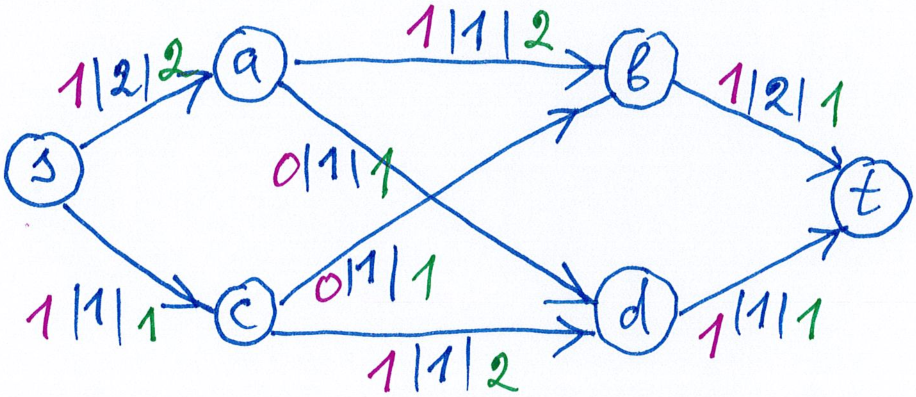
$c : A \to \mathbb{R}^+$ – les capacités des arcs

$q : A \to \mathbb{R}^+$ – les coûts des arcs

$q(\vec{uv})$ – le coût de transport d'une unité de flot sur l'arc $\vec{uv}$

Cout d'un flot f : $\text{coût}(f) = \sum_{\vec{uv} \in A} q(\vec{uv}) \cdot c(\vec{uv})$.

Exemple :



$$\text{coût}(f) = \underset{\vec{sa}}{2 \times 1} + \underset{\vec{ab}}{2 \times 1} + \underset{\vec{sc}}{1 \times 1} + \underset{\vec{ad}}{0 \times 1} + \underset{\vec{cb}}{0 \times 1} + \underset{\vec{cd}}{2 \times 1}$$

$$+ \underset{\vec{bt}}{1 \times 1} + \underset{\vec{dt}}{1 \times 1} = 9$$

But : Calculer un flot de valeur donnée (ou un flot max) et de $\boxed{\text{coût minimum.}}$

Flot de coût min comme PL :

$$
\begin{cases}
\text{minimiser} \displaystyle\sum_{\vec{uv}\in A} q_{\vec{uv}} \cdot f_{\vec{uv}} \\[2ex]
f_{\vec{uv}} \leq c_{\vec{uv}} \quad \forall\, \vec{uv}\in A \\[2ex]
\displaystyle\sum_{\vec{vt}\in\Delta^+(t)} f_{\vec{vt}} - \sum_{\vec{tv}\in\Delta^-(t)} f_{\vec{tv}} = val \\[2ex]
\displaystyle\sum_{\vec{uv}\in\Delta^+(t)} f_{\vec{uv}} - \sum_{\vec{vu}\in\Delta^-(t)} f_{\vec{vu}} = 0 \quad \forall\, v\in V\setminus\{s,t\} \\[2ex]
f_{\vec{uv}} \geq 0 \quad \forall\, \vec{uv}\in A
\end{cases}
$$

la valeur du flot voulu

De façon similaire on peut écrire en programmation lineaire :

(a) flot de coût min avec bornes inf l et sup c des capacites $(N=(V,A,l,c,q))$

(b) circulation de coût min avec demandes et bornes inf et sup sur les capacites $(N=(V,A,\underset{\text{inf}}{l},\overset{\text{sup}}{c},\underset{\text{coût}}{q},\overset{\text{demande}}{d}))$.

Le flot de coût min généralise deux problèmes classiques : le problème de transport et le problème d'affectation.

Le problème de transport : déterminer de la façon optimale l'acheminement des biens à partir de n entrepôts vers m destinataires

Exemple (exercice 2 du TD1) : 3 régions R1, R2, R3 et 3 minoteries M1, M2, M3

Distance	M1	M2	M3	Récolte
R1	210	500	400	275
R2	350	300	220	400
R3	550	200	250	300
	200	550	225	

Coût de transport : 0,1 euro/tonne × km

Cas général: n entrepôts $E_1, \dots, E_n$,

m destinataires $D_1, \dots, D_m$

a_i - la quantité du bien dans l'entrepot E_i, $i=\overline{1, \dots n}$

b_j - la démande du destinataire D_j, $j=1, \dots, m$

On suppose $\sum_{i=1}^{n} a_i = \sum_{j=1}^{m} b_j$ (tout le bien doit être transporté)

d_{ij} - la distance entre E_i et D_j

c'_{ij} - le coût de transport d'une unité du bien/km de l'entrepôt E_i vers D_j.

But: minimiser $\sum_{i=1}^{n}\sum_{j=1}^{m} c'_{ij} \cdot d_{ij} \cdot x_{ij}$ - le coût total de transport

x_{ij} - la quantité du bien transporté dépuis E_i vers D_j.

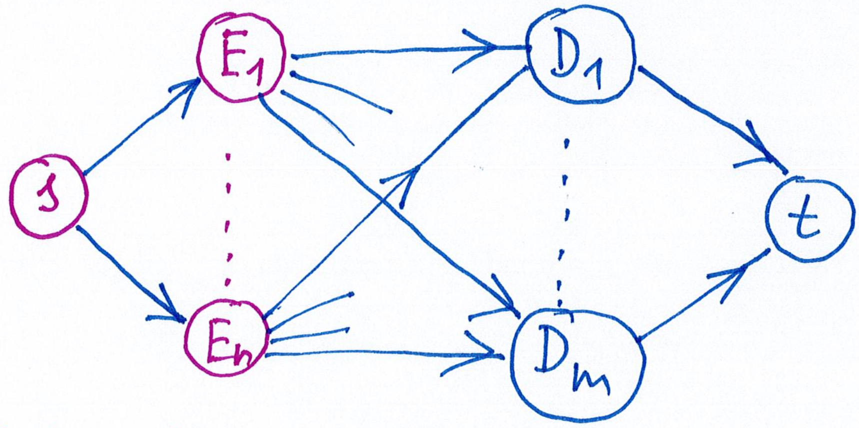
Pour simplifier, soit $c_{ij} = c'_{ij} \cdot d_{ij} \quad \forall i = 1, \ldots, n$
$\forall j = 1, \ldots, m.$

Le problème de transport comme programme lin.:

$$\begin{cases} \text{minimiser} \displaystyle\sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} x_{ij} \\ \displaystyle\sum_{j=1}^{m} x_{ij} = a_i \quad \forall i = 1, \ldots, n \\ \displaystyle\sum_{i=1}^{n} x_{ij} = b_j \quad \forall j = 1, \ldots, m \\ x_{ij} \geq 0 \quad i = 1, \ldots n, \; j = 1, \ldots m \end{cases}$$

nm variables, $n + m + nm$ contraintes

Le problème de transport comme flot de coût min



$n+m+2$ noeuds: $s, t, E_1, \ldots, E_n, D_1, \ldots D_m$

Arcs: $(s, E_i) \; \forall i$
$(D_j, t) \; \forall j$
$(E_i, D_j) \; \forall i \, \forall j$

Capacités: $c(s, E_i) = a_i \quad i = 1, \ldots n$
$c(D_j, t) = b_j \quad j = 1 \ldots m$
$c(E_i, D_j) = +\infty$

Coûts : $q(s, E_i) = 0 \quad \forall i = 1, \dots, n$

$$q(D_j, t) = 0 \quad \forall j = 1, \dots, m$$

$$q(E_i, D_j) = c_{ij} \quad \forall i = 1, \dots n \quad \forall j = 1, \dots, m$$

Calculer dans le réseau obtenu un flot de valeur $\sum a_i = \sum b_j$ (un flot max) de coût minimum.

Le problème d'affectation : n agents $A_1 \dots A_n$ et n tâches $T_1 \dots T_n$. Soit c_{ij} le prix démandé par l'agent A_i pour effectuer la tâche T_j.
Chaque agent peut réaliser une seule tâche et chaque tâche doit être réalise par un seul agent.

But : Trouver une affectation de coût total ~~opti~~ minimal.

(PLE)

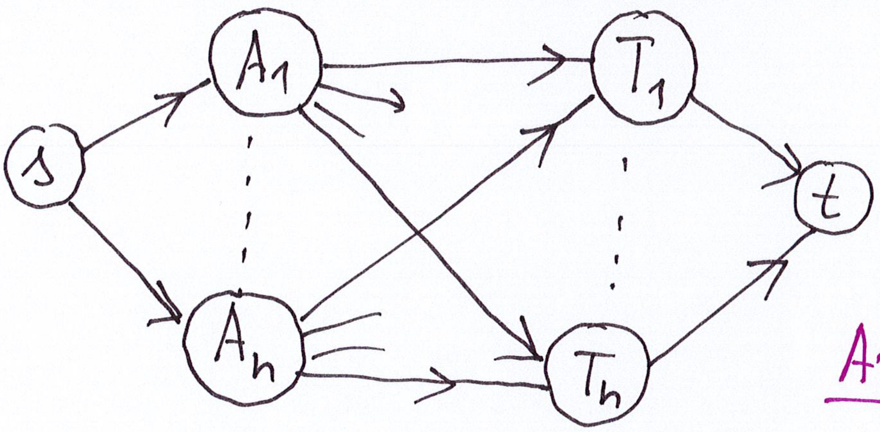
$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$$

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

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

$$x_{ij} \in \{0, 1\} \quad \forall i = 1, \dots, n, \ \forall j = 1, \dots, n$$

Affectation comme flot de coût min :



$2n+2$ noeuds

$s, t, A_1, \dots A_n, T_1, \dots, T_n$.

Arcs : (s, A_i) $\forall i$

(T_j, t) $\forall j$

(A_i, T_j) $\forall i \forall j$

Capacités : $c(s, A_i) = 1 = c(T_j, t)$
$\forall i \qquad \forall j$

$c(A_i, T_j) = 1$ (ou $+\infty$) $\forall i \forall j$

Coûts : $q(s, A_i) = 0 = q(T_j, t)$
$\forall i \qquad \forall j$

$q(A_i, T_j) = c_{ij} \quad \forall i = 1, \dots, n \quad \forall j = 1, \dots n$

Le flot de cout min peut être résolu par un algorithme combinatoire, similaire à l'algo Ford-Fulkerson (algo Busaker-Gower)

Idée: Partir d'un flot de coût minimum d'une valeur donnée (disons le flot nul)
Augmenter ce flot de façon qu'il reste un flot de cout minimum.

Soit f - le flot de coût min courant
Soit N_f - le réseau résiduel de f

Ponderation de N_f :

$$q_f(\vec{uv}) = q(\vec{uv}) \quad \forall \vec{uv} \in A \text{ t.q. } f(\vec{uv}) < c(\vec{uv})$$

$$q_f(\vec{uv}) = -q(\vec{vu}) \quad \forall \vec{vu} \in A \text{ t.q. } f(\vec{vu}) > 0.$$

Ensuite dans N_f pondéré par q_f, calculer un plus court chemin P de s à t (utiliser l'algo de Warshall-Floyd).
Augmenter f en utilisant P.

D'autres problèmes de flot exprimables en PL :

- multiflots : dans le réseau circule k types de flot entre les sources s_i et les puits t_i, $i=1,\dots,k$.

- flot de coût minimum généralisé : chaque sommet v possède aussi une demande de flot $b(v)$, qui répresente la quantité totale de flot passant par v.

- flots avec multiplicateurs : une unité de flot passant par un arc $\overrightarrow{uv}$ peut résulter dans un quantité $\mu(\overrightarrow{uv})>1$ (on gagne du flot) ou une quantité $\mu(\overrightarrow{uv})<1$ (on perd du flot) quand il arrive à l'extremité v de l'arc,