Introduction au problème du flot maximum

Renaud Sirdey (*renaud.sirdey@cea.fr*)
Commissariat à l'énergie atomique, Saclay
Cours d'optimisation combinatoire (RO03), UTC

Objectifs de ce cours

> Introduire les notions de réseau de transport et de flot.

> Introduire la notion de flot complet sur un réseau de transport.

> Poser le problème du flot maximum et montrer comment le résoudre à l'aide de la programmation linéaire et de l'algorithme de Ford-Fulkerson.

> Les problèmes de flot (plutôt dans leurs versions pondérées) ont de nombreuses applications industrielles que nous verrons dans un prochain cours.

2

Notions de réseau de transport et de *s*-*t*-flot

> Un *réseau de transport* $R=(G,u,s,t)$ consiste en la donnée :
- D’un graphe orienté $G=(V,A)$ (sans boucle).
- D’une *fonction de capacité* $u : A \rightarrow \mathbb{R}^+$.
- De deux sommets particuliers s et t de V (resp. appelés *source* et *puit*).

> Soit $R=(G,u,s,t)$ un réseau de transport, on appelle s-t-flot sur R une fonction $f : A \rightarrow \mathbb{R}^+$ telle que :
- $f(a) \leq u(a)$, pour tout $a \in A$.
- $\sum_{a \in \delta^-(v)} f(a) = \sum_{a \in \delta^+(v)} f(a)$, pour tout $v \in V \setminus \{s,t\}$ (contraintes de Kirchoff).
  - On peut poser $(t,s) \in A$ et $u((t,s)) = \infty$ pour étendre leur validité à V.
- On pose valeur$(f) = \sum_{a \in \delta^+(s)} f(a) - \sum_{a \in \delta^-(s)} f(a)$.

3

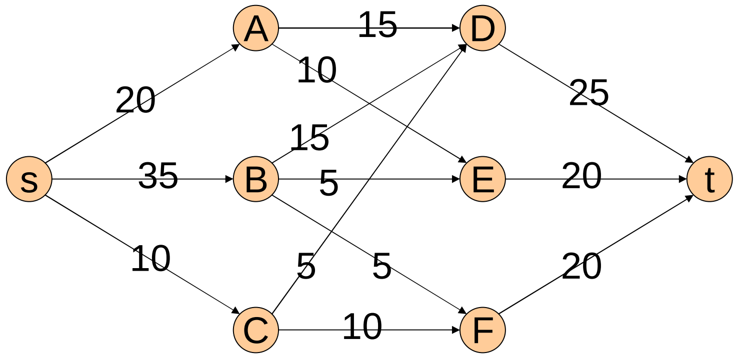
Exemple

> En trois dépôts A, B, C, on dispose respectivement de 20, 35 et 10 tonnes de marchandises. On a des demandes de 25, 20 et 20 tonnes aux destinations D, E et F. Il existe des possibilités de transport à l'aide de camions. Ces possibilités sont rapportées dans le tableau suivant :

	D	E	F
A	15	10	0
B	15	5	5
C	5	0	10

4

Réseau de transport associé



Notion de *s*-*t*-flot complet, construction

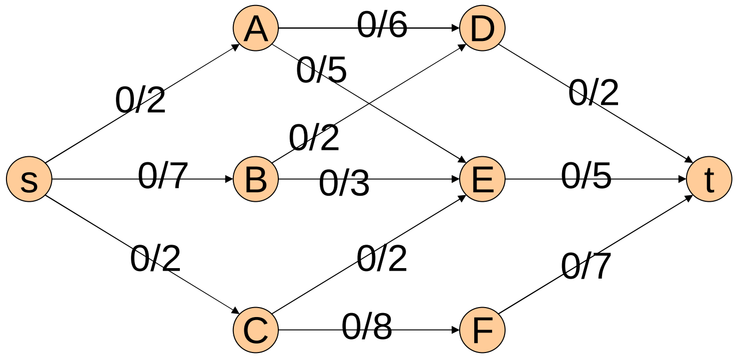
> Soit $R=(G,u,s,t)$ un réseau de transport, un *s*-*t*-flot *f* sur *R* est dit *complet* si pour tout $P=((s,v_1),(v_1,v_2),\ldots,(v_{n-1},v_n),(v_n,t))$, un *s*-*t*-chemin, il existe $a \in P$ tel que $f(a) = u(a)$.

- Autrement dit, tout chemin allant de *s* à *t* possède au moins un arc saturé par le flot.
- Un flot complet n'est pas nécessairement maximum.

> Construction d'un *s*-*t*-flot complet :

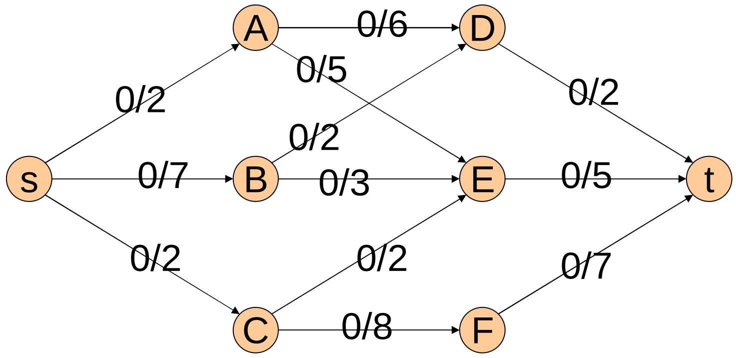
- Poser $f := 0$.
- Répéter
  - Choisir un *s*-*t*-chemin *P* tel que $\forall a \in P$ on a $f(a)<u(a)$.
  - Si un tel chemin n'existe pas alors fin.
  - Sinon pour tout $a \in P$ faire $f(a) := f(a) + \min_{a \in P} u(a)\text{-}f(a)$.

Exemple



7

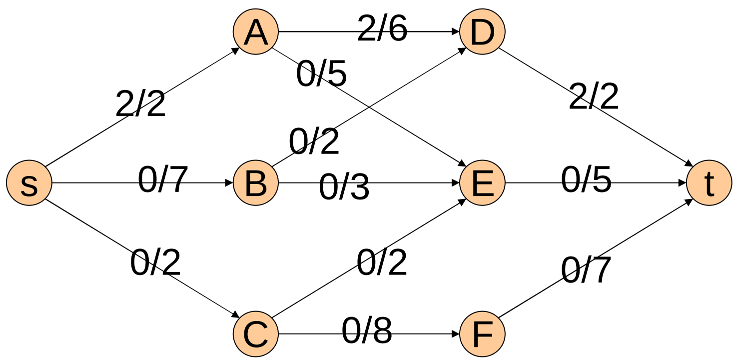
Exemple



P = ((s,A),(A,D),(D,t)), $\min_{a \in P} u(a)\text{-}f(a)=2$

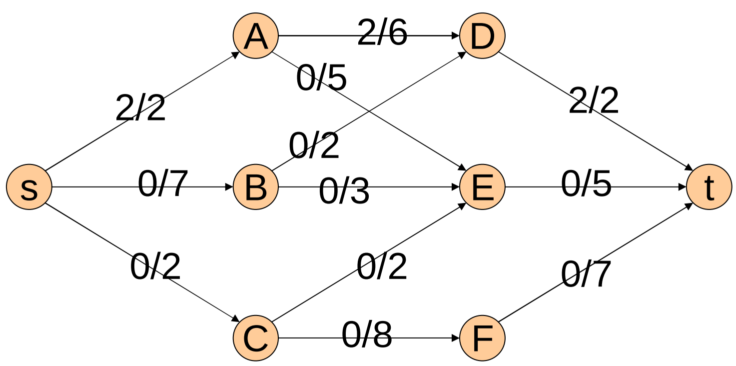
8

Exemple



9

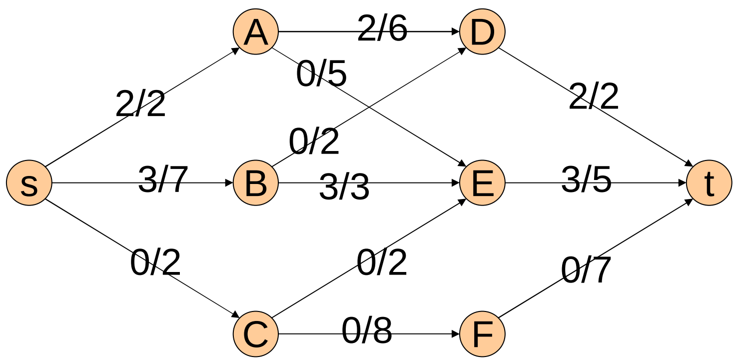
Exemple



P = ((s,B),(B,E),(E,t)), $\min_{a \in P} u(a)-f(a)=3$

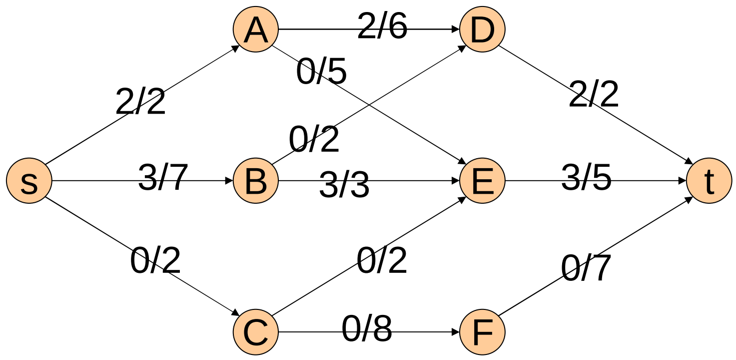
10

Exemple



11

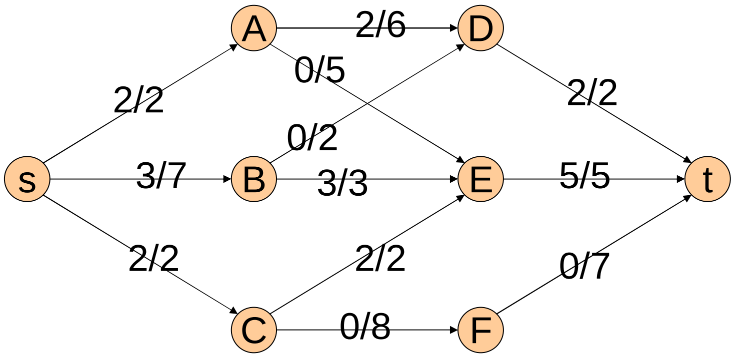
Exemple



P = ((s,C),(C,E),(E,t)), $\min_{a \in P} u(a)-f(a)=2$

12

Exemple



valeur(f) = 7

13

Le problème du *s*-*t*-flot maximum

> Soit R=(G,u,s,t) un réseau de transport, le problème du s-t-flot maximum consiste à trouver un s-t-flot f^* tel que valeur(f^*) soit maximum.

- I.e., trouver f^* t. q. pour tout s-t-flot f sur R on ait valeur(f) ≤ valeur(f^*).

14

Approche par programmation linéaire

> Formulation du problème :

Maximiser valeur(f) = $\sum_{a \in \delta^+(s)} f(a) - \sum_{a \in \delta^-(s)} f(a)$,
sous les contraintes,
$\sum_{a \in \delta^-(v)} f(a) = \sum_{a \in \delta^+(v)} f(a)$, $\forall\, v \in V \setminus \{s,t\}$,
$f(a) \leq u(a)$, $\forall\, a \in A$,
$f(a) \geq 0$, $\forall\, a \in A$.

> Il s'agit donc d'un programme linéaire : $\min\{c^T x : Ax \leq b\}$.

- Avec $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{mxn}$, $b \in \mathbb{R}^n$.

> Remarque : le système d'inégalités linéaires ci-dessus possède toujours au moins une solution.

15

Conséquences

> Le problème du s-t-flot maximum est un problème polynomial.

- Conséquence direct du théorème de Khachiyan.

> On peut utiliser n'importe quel algorithme pour la programmation linéaire (par ex. l'algorithme du simplexe) afin de résoudre ce problème.

- En termes de complexité nous allons faire mieux à l'aide d'algorithmes « combinatoires ».
- En pratique, par contre, ce n'est pas forcément idiot, mais cela devient une question de génie logiciel.

16

Notion de graphe résiduel

> Soit $G=(V,A)$ un graphe orienté (sans boucle), on définit $G'=(V,A\cup A')$ tel que si $a=(v,w)\in A$ alors $a'=(w,v)\in A'$.

- Remarque : G' peut contenir des arcs parallèles alors que G n'en contient pas.

> Soit un réseau de transport $R=(G,u,s,t)$ et un s-t-flot f sur R, on définit la *fonction de capacité résiduelle* $u_f : A\cup A'\rightarrow\mathbb{R}^+$ t. q. :

- $u_f(a) = u(a) - f(a)$, pour tout $a\in A$.
- $u_f(a') = f(a)$, pour tout $a=(v,w)\in A$ et $a'=(w,v)\in A'$.

> On appelle *graphe résiduel associé à f* le graphe

$$G_f = (V,\{a\in A\cup A' : u_f(a) > 0\}).$$

17

Notion de chemin *f*-augmentant

> Soit $R=(G,u,s,t)$ un réseau de transport, f un s-t-flot sur R, P un chemin (ou un circuit) sur G_f et $\gamma > 0$, *augmenter f* de γ selon P signifie faire :

- $f(a) := f(a) + \gamma$ si $a \in A$.
- $f(a) := f(a) - \gamma$ si $a' \in A'$.

> On appelle *chemin f-augmentant*, tout s-t-chemin dans le graphe résiduel G_f.

18

L'algorithme de Ford-Fulkerson

> Entrée : $R=(G,u,s,t)$ un réseau de transport.

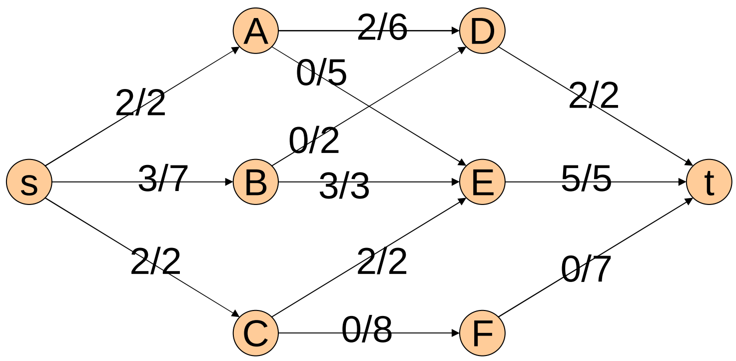
> Sortie : un s-t-flot f de valeur maximum.

> Énoncé :

- Poser $f := 0$.
- Répéter :
  - Choisir un chemin f-augmentant P.
  - Si un tel chemin n'existe pas alors fin (f est alors maximum).
  - Augmenter f de $\gamma=\min_{a\in P} u_f(a)$ selon P.

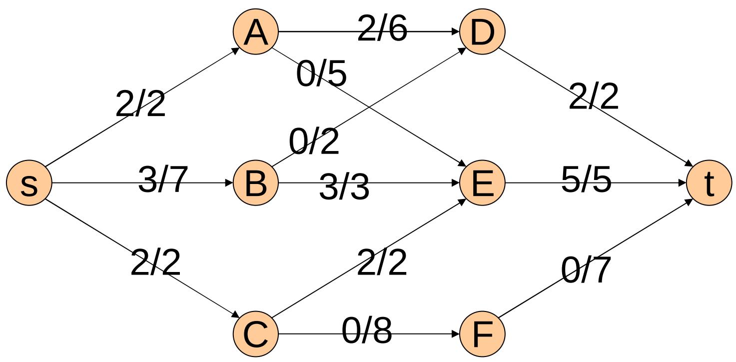
19

Exemple (précédent)



20

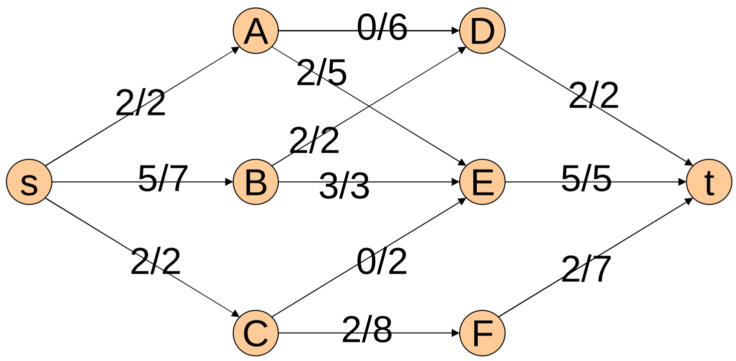
Exemple (précédent)



P = ((s,B),(B,D),(D,A),(A,E),(E,C),(C,F),(F,t)), $\min_{a \in P} u_f(a) = 2$

21

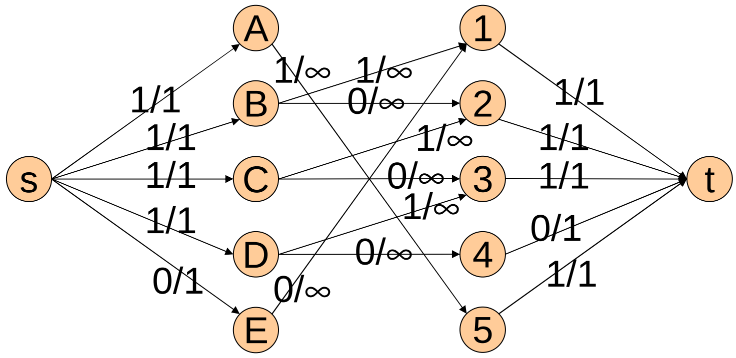
Exemple (précédent)



valeur(f) = 9, f est maximum

22

À vous…

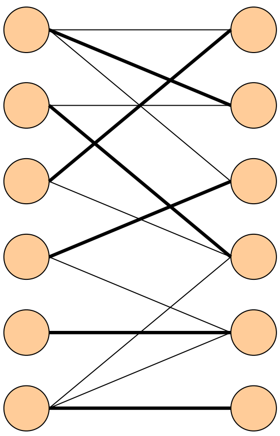


Transformez le flot complet ci-dessus en flot maximum.

23

Retour sur le problème du couplage biparti de cardinalité maximum

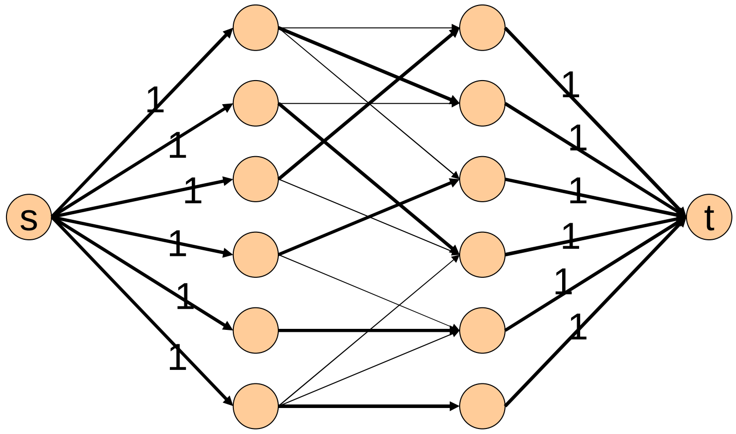
> Mettre ce problème sous la forme d'un problème de flot maximum.



24

Retour sur le problème du couplage biparti de cardinalité maximum

> Mettre ce problème sous la forme d'un problème de flot maximum.



25

Lemme des contraintes de Kirchoff généralisées

> Soit R=(G,u,s,t) un réseau de transport et f un s-t-flot sur R alors pour tout $X \subseteq V \setminus \{s,t\}$ on a :

$$\sum_{a \in \delta^-(X)} f(a) = \sum_{a \in \delta^+(X)} f(a).$$

> Preuve :

Pour tout $v \in X$ on a $\sum_{a \in \delta^-(v)} f(a) - \sum_{a \in \delta^+(v)} f(a) = 0$,
d'où $\sum_{v \in X} (\sum_{a \in \delta^-(v)} f(a) - \sum_{a \in \delta^+(v)} f(a)) =0$.
Or $\delta^-(v)= \delta^-(v) \cap \delta^-(X) \cup \delta^-(v) \setminus \delta^-(X)$ et $\delta^-(v)= \delta^+(v) \cap \delta^+(X) \cup \delta^+(v) \setminus \delta^+(X)$,
$\sum_{v \in X} (\sum_{a \in \delta^-(v) \cap \delta^-(X)} f(a) - \sum_{a \in \delta^+(v) \cap \delta^+(X)} f(a)) = \sum_{a \in \delta^-(X)} f(a) - \sum_{a \in \delta^+(X)} f(a)$
et $\sum_{v \in X} (\sum_{a \in \delta^-(v) \setminus \delta^-(X)} f(a) - \sum_{a \in \delta^+(v) \setminus \delta^+(X)} f(a)) = \sum_{a \in A(X)} f(a) - \sum_{a \in A(X)} f(a) = 0$.
□

26

Lemme

> Soit $R=(G,u,s,t)$ un réseau de transport et f un s-t-flot sur R alors pour tout $X \subseteq V$ t. q. $s \in X$ et $t \notin X$ on a :

(a) valeur(f) = $\sum_{a \in \delta^+(X)} f(a) - \sum_{a \in \delta^-(X)} f(a)$.

(b) valeur(f) ≤ $\sum_{a \in \delta^+(X)} u(a)$.

> Preuve :

(a) : analogue à la preuve du lemme des contraintes de Kirchoff généralisées.

(b) : conséquence triviale de (a) et de $0 \leq f(a) \leq u(a)$.

□

27

Théorème

> Soit $R=(G,u,s,t)$ un réseau de transport, un s-t-flot f sur R est maximum si et seulement si il n'existe pas de chemin f-augmentant.

> Preuve :

Nécessité : s'il existe un chemin f-augmentant alors l'algorithme de Ford-Fulkerson fourni un flot de plus grande valeur et donc f n'est pas maximum.

Suffisance : s'il n'y a pas de chemin f-augmentant t n'est (par définition) pas atteignable depuis s dans G_f. Soit Y l'ensemble des sommets atteignables depuis s dans G_f. Par définition de G_f on a $f(a)=u(a)$ (car $u_f(a)=u(a)-f(a)=0$) pour tout a de $\delta^+_G(Y)$ et $f(a)=0$ (car $u_f(a')=f(a)=0$) pour tout a de $\delta^-_G(Y)$. Le lemme précédent (partie (a)) implique

valeur(f) = $\sum_{a \in \delta^+_G(Y)} u(a)$,

et la partie (b) permet de conclure.

□

28

Lien avec le problème de la coupe minimum

> Soit $G=(V,A)$ un graphe orienté, $C \subseteq A$ est une s-t-coupe sur G si $G=(V,A \setminus C)$ ne contient pas de s-t-chemin.

- On définit également capa(C) = $\sum_{a \in C} u(a)$.

> Théorème : soit $R=(G,u,s,t)$ un réseau de transport, la valeur maximum d'un s-t-flot sur R est égal à la capacité minimum d'une s-t-coupe.

> Preuve :

Dans la preuve du théorème précédent nous avons construit une s-t-coupe, $\delta^+(Y)$, dont la capacité est égale à la valeur d'un flot maximum, f.

À toute s-t-coupe minimale C on peut associer $X \subseteq V$ t. q. $s \in X$, $t \notin X$ et $C=\delta^+(X)$

Or (lemme précédent), on a capa($\delta^+(X)$) = $\sum_{a \in \delta^+(X)} u(a) \geq$ valeur(f).

□

29

Remarques sur l'intégrité

> Corollaire : soit $R=(G,u,s,t)$ un réseau de transport avec $u : A \rightarrow \mathbb{Z}^+$, alors il existe un s-t-flot maximum à valeurs entières et l'algorithme de Ford-Fulkerson termine avec un tel flot.

> Preuve :

Lorsque la fonction de capacité est à valeurs entières $\gamma=\min_{a\in P} u_f(a)$ est entier.

□

> Remarque : c'est aussi le cas des solutions du PL précédent car les matrices associées aux problèmes de flot sont totalement unimodulaires (i.e., tous leurs sous-déterminants sont 0, +1 ou -1).

30

Remarques algorithmiques

> Pour obtenir des algorithmes efficaces, il convient de ne pas choisir le chemin *f*-augmentant arbitrairement.

- Par ex. avec des capacités irrationnelles, l'algorithme tel que nous l'avons décrit ne termine pas forcément…

> Par ex. dans l'algorithme d'Edmonds-Karp on choisit le plus court des chemins *f*-augmentant.

- Théorème : l'algorithme d'Edmonds-Karp résout le problème du s-t-flot maximum en $O(m^2n)$.

> D'autres algorithmes existent, par ex. l'algorithme de Golberg-Tarjan qui résout le problème en $O(n^3)$.

31