

Cours 1 - Algorithmes Polynomiaux

Flot Maximum et Coupe Minimum

Semestre Automne 2020-2021 - Université Claude Bernard Lyon 1

Christophe Crespelle

`christophe.crespelle@inria.fr`



département

Informatique

Faculté des Sciences et Technologies

Université Claude Bernard Lyon 1

Optimisation et Recherche Operationnelle

- Qu'est-ce que c'est ?
 - ▶ **Recherche operationnelle (wiki)** : recherche du meilleur choix dans la façon d'opérer en vue d'aboutir au résultat visé ou au meilleur résultat possible.
 - ▶ **Optimisation combinatoire** : recherche de la meilleure solution (fonction objectif) a un probleme qui admet un grand nombre (fini) de solutions possibles

Optimisation et Recherche Operationnelle

- Qu'est-ce que c'est ?
 - ▶ **Recherche operationnelle (wiki)** : recherche du meilleur choix dans la façon d'opérer en vue d'aboutir au résultat visé ou au meilleur résultat possible.
 - ▶ **Optimisation combinatoire** : recherche de la meilleure solution (fonction objectif) a un probleme qui admet un grand nombre (fini) de solutions possibles
- A quoi ca sert ?
 - ▶ gagner plus d'argent. ex : maximiser la production, minimiser les ressources utilisees
 - ▶ ameliorer la qualite d'un service. ex : ou placer les points de services, recommander des contenus, minimiser le temps de reponse
 - ▶ reconstruire des donnees manquantes. ex : sequencer le genome

Optimisation et Recherche Operationnelle

- Qu'est-ce que c'est ?
 - ▶ **Recherche operationnelle (wiki)** : recherche du meilleur choix dans la façon d'opérer en vue d'aboutir au résultat visé ou au meilleur résultat possible.
 - ▶ **Optimisation combinatoire** : recherche de la meilleure solution (fonction objectif) a un probleme qui admet un grand nombre (fini) de solutions possibles
- A quoi ca sert ?
 - ▶ gagner plus d'argent. ex : maximiser la production, minimiser les ressources utilisees
 - ▶ ameliorer la qualite d'un service. ex : ou placer les points de services, recommander des contenus, minimiser le temps de reponse
 - ▶ reconstruire des donnees manquantes. ex : sequencer le genome
- Domaines et contextes d'application :
 - ▶ gestion de projets
 - ▶ industrie
 - ▶ finance
 - ▶ energie
 - ▶ informatique

Exemple 1 : implantation d'hopitaux

- **Donnee** : une densite de population, k hopitaux a placer

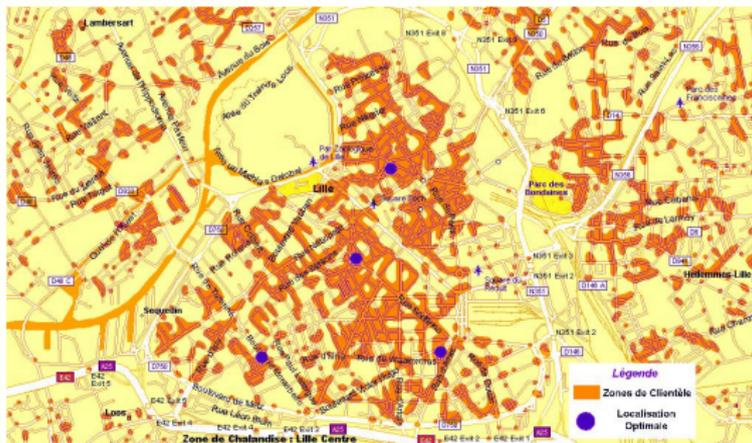


FIGURE – Un probleme d'implantation avec 4 hopitaux.

- **Probleme 1** : placer les 4 hopitaux de sorte a minimiser la distance *maximale* d'un foyer de population a l'hopital le plus proche

Exemple 1 : implantation d'hopitaux

- **Donnee** : une densite de population, k hopitaux a placer

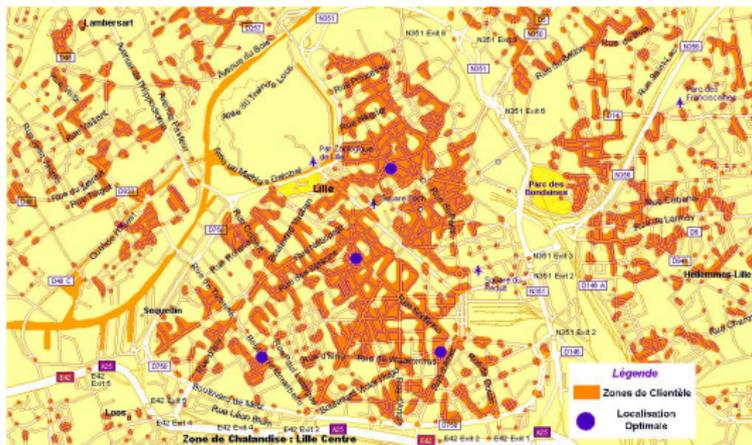


FIGURE – Un probleme d'implantation avec 4 hopitaux.

- **Probleme 1** : placer les 4 hopitaux de sorte a minimiser la distance *maximale* d'un foyer de population a l'hopital le plus proche
- **Probleme 2** : minimiser la distance *moyenne* d'un foyer de population a l'hopital le plus proche

Exemple 2 : equilibrage de charges

- **Donnee** : un ensemble de taches T_1, T_2, \dots, T_n , chacune avec une duree $d(T_i)$, et k postes de travail



FIGURE – Un probleme d'equilibrage de charges avec 12 taches et 4 postes de travail.

- **Probleme** : trouver une affectation des taches sur les postes qui minimise le temps de fin d'execution

Objectifs du cours

- **Differents types de problemes d'optimisation**

flot maximum, coupe minimum, stable maximum, voyageur de commerce, equilibrage de charges, sac a dos, couverture par sommets, selection de k centres

- **Techniques algorithmiques generales pour les resoudre**

- ▶ algorithmes polynomiaux solution exacte
- ▶ algorithmes probabilistes solution **souvent** optimale
- ▶ branching solution exacte
- ▶ programmation dynamique solution exacte
- ▶ algorithmes d'approximation solution **proche** de l'optimal
- ▶ programmation lineaire et en nombre entiers solution exacte
- ▶ (meta-)heuristiques **"bonne"** solution

Objectifs du cours

- **Differents types de problemes d'optimisation**
flot maximum, coupe minimum, stable maximum, voyageur de commerce, equilibrage de charges, sac a dos, couverture par sommets, selection de k centres
- **Techniques algorithmiques generales pour les resoudre**
 - ▶ algorithmes polynomiaux
 - ▶ algorithmes probabilistes
 - ▶ branching
 - ▶ programmation dynamique
 - ▶ algorithmes d'approximation
 - ▶ programmation lineaire et en nombre entiers
 - ▶ (meta-)heuristiques

Complexite polynomiale : traite de grandes instances

Complexite exponentielle : traite seulement de petites instances

Organisation pratique du cours

Volume :

- CM : 15h
- TD : 9h
- TP : 6h

Evaluation (en controle continu partiel) : **NON CONTRACTUEL ;)**

- examen final : 60%
- test intermediaire : 15%
- evaluation des TP : 25%
 - ▶ un rendu en fin de seance
 - ▶ un rendu en fin de semestre (mini-projet=finir le TP1)

Emploi du temps (Aie!) :

<https://perso.ens-lyon.fr/christophe.crespelle/enseignements.html>

- alternance bleu/jaune (**Attention !!! 7 jours inverses**)
- CM en comodal
- TD en presence pure + une session a distance pour tout le monde
- TP une seance en presence et une a distance

Reussir l'UE

Comment reussir l'UE ?

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

- Poser des questions en CM

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

- Poser des questions en CM
- Poser des questions en CM

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

- Poser des questions en CM
- Poser des questions en CM
- Revoir les notions de base dans *Introduction a l'algorithmique*, Cormen et al.

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

- Poser des questions en CM
- Poser des questions en CM
- Revoir les notions de base dans *Introduction a l'algorithmique*, Cormen et al.
- Retravailler les CM a posteriori

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

- Poser des questions en CM
- Poser des questions en CM
- Revoir les notions de base dans *Introduction a l'algorithmique*, Cormen et al.
- Retravailler les CM a posteriori
- Travailler les corrections des TD

Reussir l'UE

Comment reussir l'UE ?

- Apprendre les CM par coeur
- Finir les 2 TP
- Savoir faire les TD (corrections integrales fournies)

Que faire si vous etes perdu ?

- Poser des questions en CM
- Poser des questions en CM
- Revoir les notions de base dans *Introduction a l'algorithmique*, Cormen et al.
- Retravailler les CM a posteriori
- Travailler les corrections des TD
- Consultez les passages de livre sur le cours (versions electroniques)
 - ▶ *Introduction a l'algorithmique*, Cormen et al.
 - ▶ *Algorithm Design*, Kleinberg et Tardos
 - ▶ *Exact Exponential Algorithms*, Fomin et Kratsch

I. Problème du flot maximum

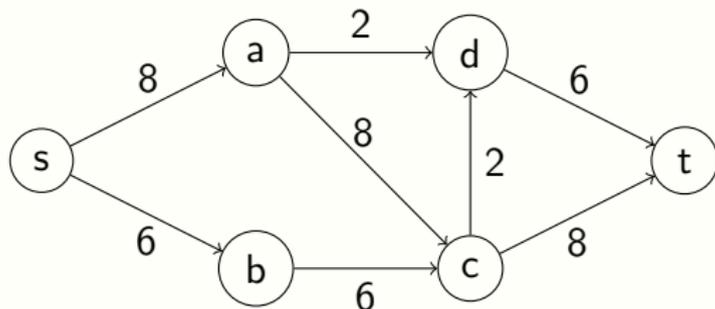
- **Entrée :**

- ▶ un graphe orienté $G = (V, A)$ **sans arcs symétriques**
- ▶ une fonction de capacité $c : \begin{array}{l} A \rightarrow \mathbb{R}^{+*} \\ (u, v) \mapsto c(u, v) \end{array}$
- ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)

I. Problème du flot maximum

- **Entrée : un reseau de flot (flow network en anglais)**

- ▶ un graphe orienté $G = (V, A)$ sans arcs symétriques
- ▶ une fonction de capacité $c : A \rightarrow \mathbb{R}^{+*}$
 $(u, v) \mapsto c(u, v)$
- ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)
 s est la source, t est le puits (*tank* en anglais)



I. Problème du flot maximum

- **Entrée : un réseau de flot (flow network en anglais)**

- ▶ un graphe orienté $G = (V, A)$ sans arcs symétriques
- ▶ une fonction de capacité $c : A \rightarrow \mathbb{R}^{+*}$
 $(u, v) \mapsto c(u, v)$
- ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)
 s est la source, t est le puits (*tank* en anglais)

Définition (flot et valeur d'un flot)

Un *flot* est une fonction $f : A(G) \rightarrow \mathbb{R}^+$ qui satisfait les deux conditions suivantes :

- **Contraintes de capacité :** $\forall (u, v) \in A, 0 \leq f(u, v) \leq c(u, v)$
- **Conservation :**

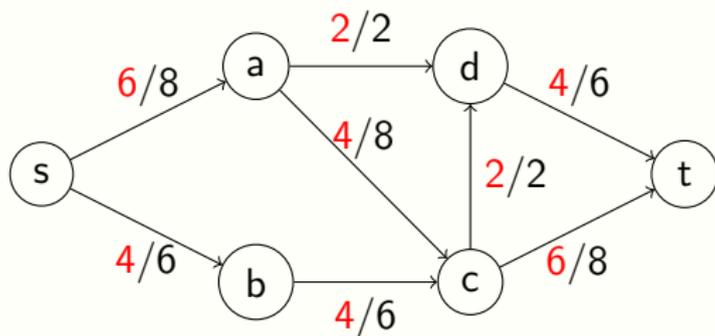
$$\forall u \in V \setminus \{s, t\}, \sum_{v \in N^-(u)} f(v, u) = \sum_{v \in N^+(u)} f(u, v)$$

La *valeur d'un flot* f , notée $|f|$, est la quantité relative de flot qui sort de s : $|f| = \sum_{v \in N^+(s)} f(s, v) - \sum_{v \in N^-(s)} f(v, s)$.

Ou de manière équivalente $|f| = \sum_{v \in N^-(t)} f(v, t) - \sum_{v \in N^+(t)} f(t, v)$.

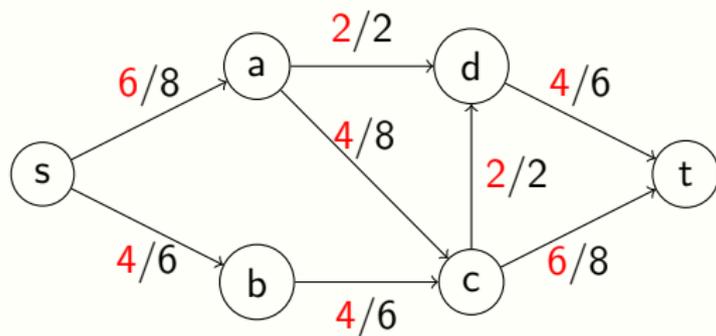
I. Problème du flot maximum

Un flot f



I. Problème du flot maximum

Un flot f



de valeur $|f| = 10$.

I. Problème du flot maximum

- **Entrée : un réseau de flot (flow network en anglais)**
 - ▶ un graphe orienté $G = (V, A)$ **sans arcs symétriques**
 - ▶ une fonction de capacité $c : A \rightarrow \mathbb{R}^{+*}$
 $(u, v) \mapsto c(u, v)$
 - ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)

I. Problème du flot maximum

- **Entrée : un reseau de flot (flow network en anglais)**
 - ▶ un graphe orienté $G = (V, A)$ sans arcs symetriques
 - ▶ une fonction de capacite $c : A \rightarrow \mathbb{R}^{+*}$
 $(u, v) \mapsto c(u, v)$
 - ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)
- **Sortie : un flot sur G de valeur maximum**

I. Problème du flot maximum

- **Entrée : un reseau de flot (flow network en anglais)**
 - ▶ un graphe orienté $G = (V, A)$ **sans arcs symetriques**
 - ▶ une fonction de capacite $c : A \rightarrow \mathbb{R}^{+*}$
 $(u, v) \mapsto c(u, v)$
 - ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)
- **Sortie : un flot sur G de valeur maximum**

On notera $|f^*|$ la valeur maximum d'un flot et f^* un flot qui realise cette valeur.

I. Problème du flot maximum

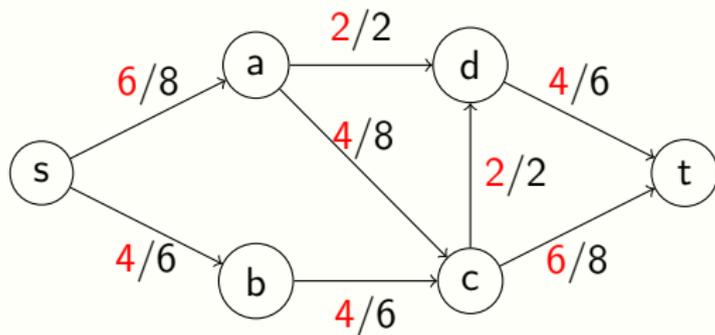
- **Entrée : un reseau de flot (flow network en anglais)**
 - ▶ un graphe orienté $G = (V, A)$ sans arcs symetriques
 - ▶ une fonction de capacite $c : A \rightarrow \mathbb{R}^{+*}$
 $(u, v) \mapsto c(u, v)$
 - ▶ un couple $(s, t) \in V^2$ de sommets distincts ($s \neq t$)
- **Sortie : un flot sur G de valeur maximum**

On notera $|f^*|$ la valeur maximum d'un flot et f^* un flot qui realise cette valeur.

Question : $|f^*|$ (et donc f^*) est elle correctement definie ?

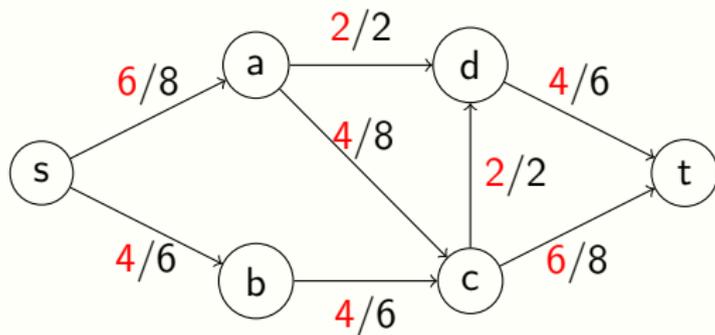
I. Problème du flot maximum

f est-il un flot maximal ?



I. Problème du flot maximum

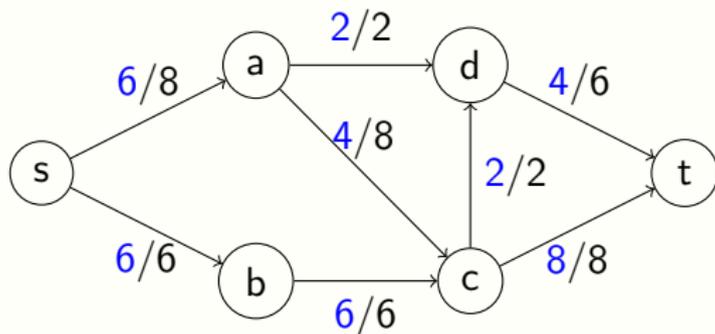
f est-il un flot maximal? \rightarrow NON



I. Problème du flot maximum

f est-il un flot maximal? \rightarrow NON

f' est maximal ($|f'| = 12$).

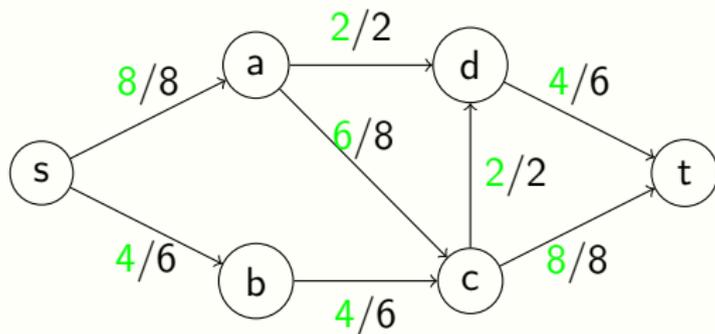


I. Problème du flot maximum

f est-il un flot maximal? \rightarrow NON

f' est maximal ($|f'| = 12$).

f'' est aussi maximal ($|f''| = 12$).



Cadre plus general pour l'entree

- Arcs symetriques ?

Cadre plus general pour l'entree

- Arcs symetriques ? (au fait, utile?)

Cadre plus general pour l'entree

- Arcs symetriques ? (au fait, utile?)
- Graphes non orientes ?

Cadre plus general pour l'entree

- Arcs symetriques ? (au fait, utile?)
- Graphes non orientes ?
- Capacites nulles ?

Cadre plus general pour l'entree

- Arcs symetriques ? (au fait, utile?)
- Graphes non orientes ?
- Capacites nulles ?
- Sources et puits multiples ?

Cadre plus general pour l'entree

- Arcs symetriques ? (au fait, utile ?)
- Graphes non orientes ?
- Capacites nulles ?
- Sources et puits multiples ?

Questions : Quel temps de calcul prennent ces transformations ?
Penalisent elles la complexite de l'algorithme ?

II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)

II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)

Définition (s, t -coupe et sa capacite)

Une s, t -coupe est une bipartition (S, T) de V telle que $s \in S$ et $t \in T$.

La *capacite* de la coupe (S, T) est $C(S, T) = \sum_{(u,v) \in (S \times T) \cap A} c(u, v)$

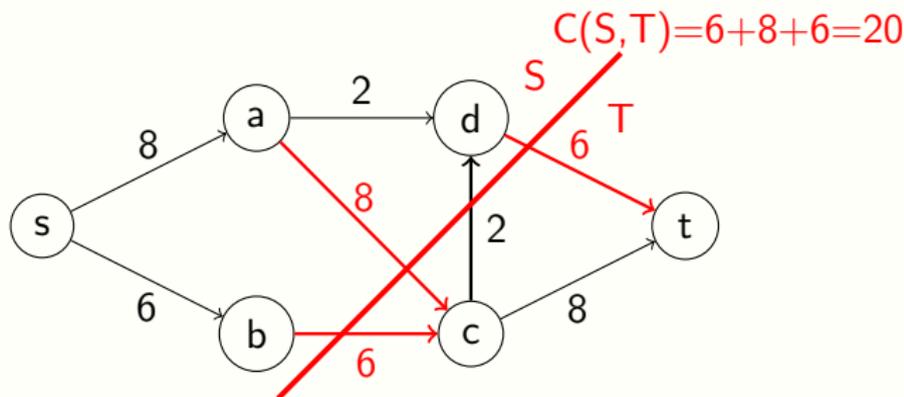
II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)

Définition (s, t -coupe et sa capacite)

Une s, t -coupe est une bipartition (S, T) de V telle que $s \in S$ et $t \in T$.

La *capacite* de la coupe (S, T) est $C(S, T) = \sum_{(u,v) \in (S \times T) \cap A} c(u, v)$



II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)

II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)
- **Sortie** : une s, t -coupe de G de capacite **minimum**

II. Probleme de la coupe minimum

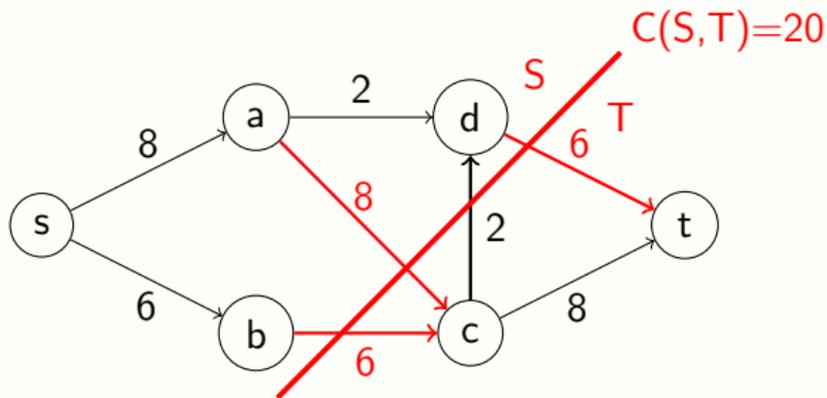
- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)
- **Sortie** : une s, t -coupe de G de capacite **minimum**

Question : la capacite *minimum* d'une s, t -coupe est elle correctement definie ?

II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)
- **Sortie** : une s, t -coupe de G de capacite **minimum**

(S,T) est-elle une coupe minimum ?

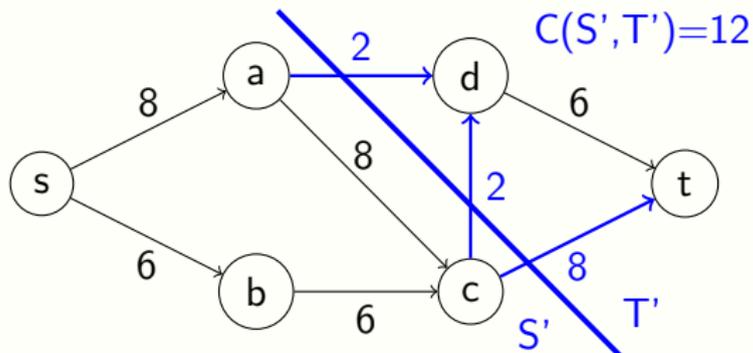


II. Probleme de la coupe minimum

- **Entrée** : un reseau de flot $G = (V, A)$ avec une source s et un puit t (exactement comme precedemment)
- **Sortie** : une s, t -coupe de G de capacite **minimum**

(S, T) est-elle une coupe minimum ? \rightarrow NON

(S', T') a capacite 12.



III. Le theoreme flot maximum / coupe minimum

Théorème

Soit G un reseau de flot ayant pour source s et pour puits t . La capacite minimum d'une s, t -coupe de G est egale a la valeur maximum d'un flot entre s et t dans G .

Remarque

L'annonce du theoreme sous-entend qu'il existe un flot de valeur maximum, la preuve devra l'etablir.

Reseau residuel

Définition (***) Reseau residuel d'un flot f)

C'est un reseau de flot, note $G_f = (V_f, A_f)$, avec **possibilite d'arcs symetriques**, defini comme suit :

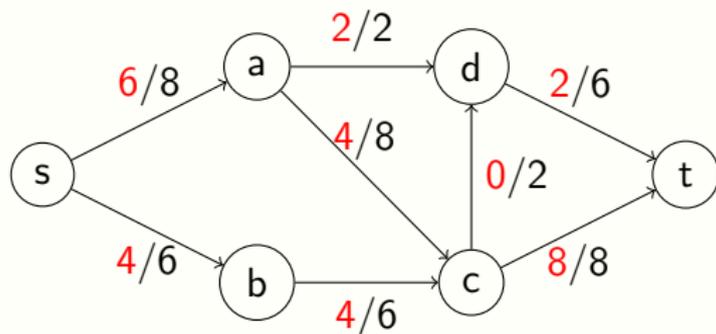
- $V_f = V$, meme source s et puits t que G
- $\forall (u, v) \in A \cup A^r$,
$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{si } (u, v) \in A \\ f(v, u) & \text{si } (v, u) \in A \end{cases}$$
- $A_f = \{(u, v) \in A \cup A^r \mid c_f(u, v) > 0\}$

Remarque

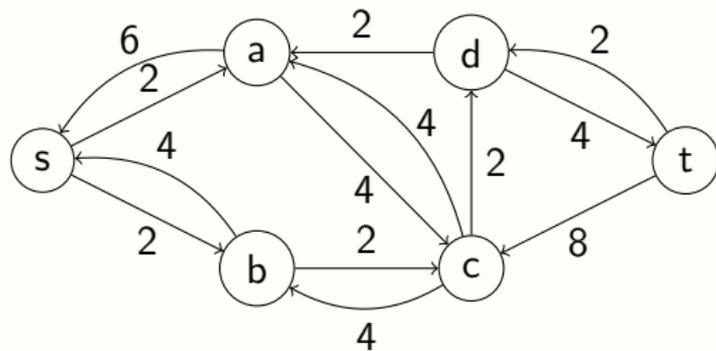
La definition d'un flot dans un reseau sans arcs symetriques est valide aussi avec possibilite d'arcs symetriques : on l'adopte pour faire des flots dans le reseau residuel.

Exemple de residuel

Un reseau de flot G et un flot f



Le residuel G_f de f



Flot dans le résiduel

Lemme

Soit f' un flot du réseau résiduel G_f de f et soit $f + f'$ défini par $\forall (u, v) \in A, (f + f')(u, v) = f(u, v) + f'(u, v) - f'(v, u)$.

(en considérant $f'(x, y) = 0$ si l'arc (x, y) n'existe pas dans G_f)

Alors, $f + f'$ est un flot sur G et sa valeur est $|f| + |f'|$.

Démonstration.

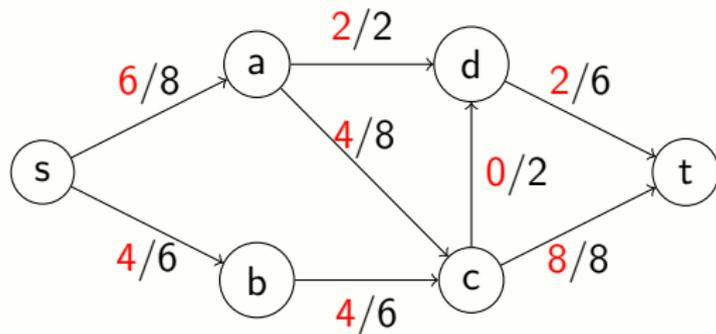
Vérifier :

- contraintes de capacité
- conservation
- valeur

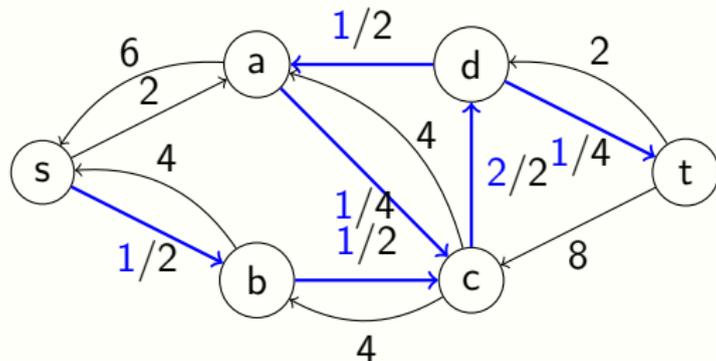


Flot dans le residuel

Un flot f dans G

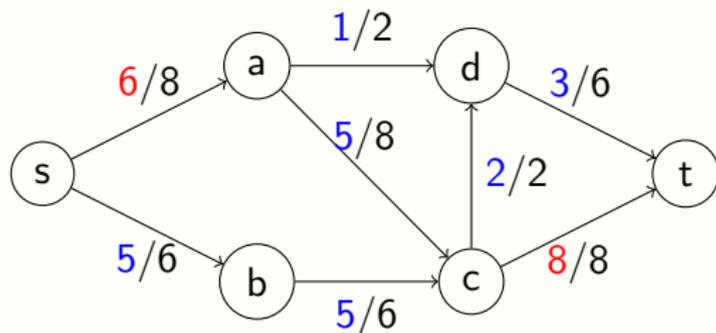


Un flot f' dans le residuel G_f

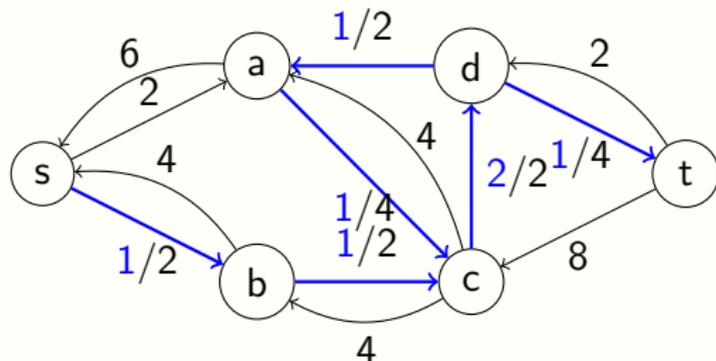


Flot dans le residuel

Le flot $\tilde{f} = f + f'$ dans G



Un flot f' dans le residuel G_f

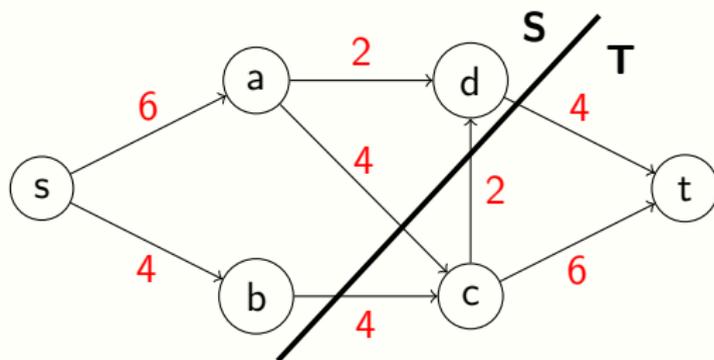


Flot net a travers une s, t -coupe

Définition

Soit $G = (V, A)$ un reseau de flot relache et (S, T) une s, t -coupe de G . Le flot net a travers (S, T) est defini par

$$f(S, T) = \sum_{(u,v) \in (S \times T) \cap A} f(u, v) - \sum_{(v,u) \in (T \times S) \cap A} f(v, u).$$

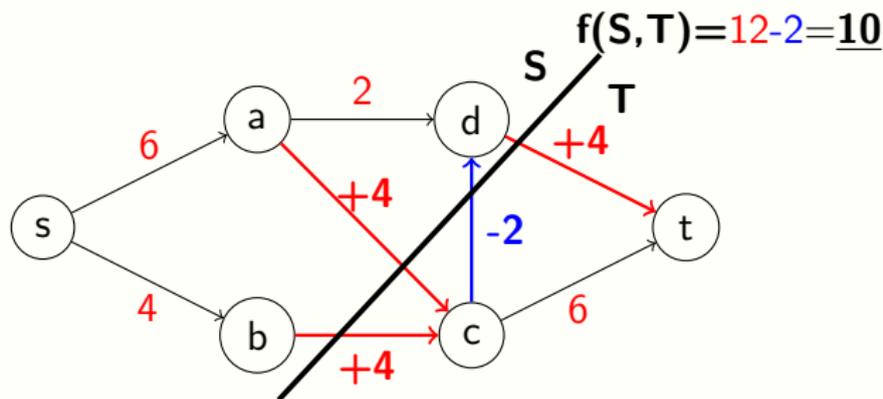


Flot net a travers une s, t -coupe

Définition

Soit $G = (V, A)$ un reseau de flot relache et (S, T) une s, t -coupe de G . Le flot net a travers (S, T) est defini par

$$f(S, T) = \sum_{(u,v) \in (S \times T) \cap A} f(u, v) - \sum_{(v,u) \in (T \times S) \cap A} f(v, u).$$



Flot net a travers une s, t -coupe

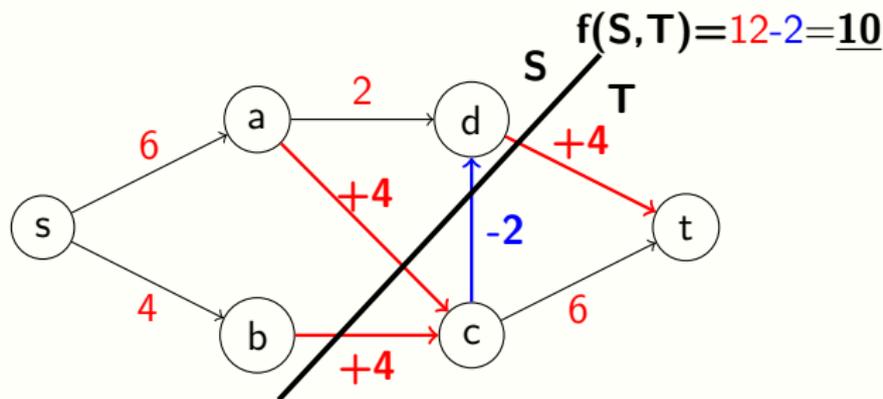
Définition

Soit $G = (V, A)$ un reseau de flot relache et (S, T) une s, t -coupe de G . Le flot net a travers (S, T) est defini par

$$f(S, T) = \sum_{(u,v) \in (S \times T) \cap A} f(u, v) - \sum_{(v,u) \in (T \times S) \cap A} f(v, u).$$

Remarque (***)

On a toujours $f(S, T) \leq C_G(S, T)$.



Flot net a travers une s, t -coupe

Définition

Soit $G = (V, A)$ un reseau de flot relache et (S, T) une s, t -coupe de G . Le flot net a travers (S, T) est defini par

$$f(S, T) = \sum_{(u,v) \in (S \times T) \cap A} f(u, v) - \sum_{(v,u) \in (T \times S) \cap A} f(v, u).$$

Remarque (***)

On a toujours $f(S, T) \leq C_G(S, T)$.

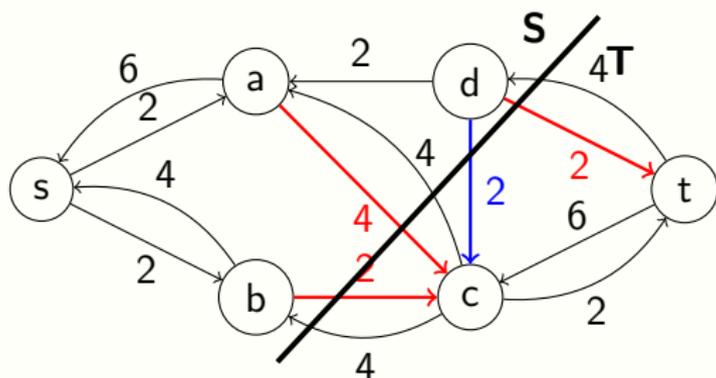
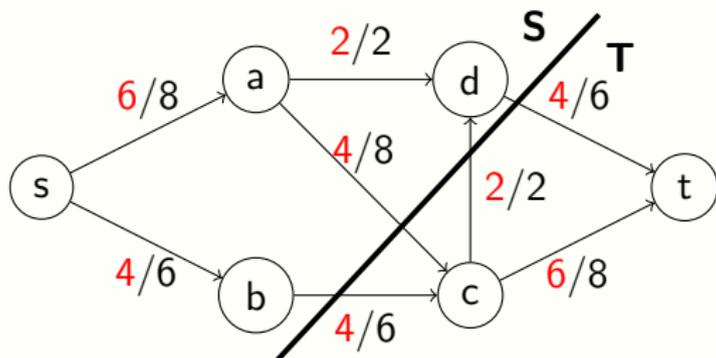
Remarque

La capacite de la s, t -coupe (S, T) dans le reseau residuel G_f verifie $C_{G_f}(S, T) = C_G(S, T) - f(S, T)$.

Capacite d'une coupe dans le residuel

$$C_{G_f}(S, T) = C_G(S, T) - f(S, T) = C_G(S, T) - (f_{out}(S, T) - f_{in}(S, T))$$

$$= (C_G(S, T) - f_{out}(S, T)) + f_{in}(S, T)$$



Chemin augmentant dans G_f et sa capacité résiduelle

Définition

Soit G un réseau de flot, f un flot sur G et G_f le résiduel de f . Un chemin augmentant P est un chemin de s à t dans G_f . La capacité résiduelle de P est défini par $c_f(P) = \min\{c_f(u, v) \mid (u, v) \in P\}$.

Remarque

Soit P un chemin augmentant dans G_f et soit f_P défini par

$$\forall (u, v) \in A_f, \quad f_P(u, v) = \begin{cases} c_f(P) & \text{si } (u, v) \in P \\ 0 & \text{sinon} \end{cases} .$$

Alors, f_P est un flot dans G_f et sa valeur est $|f_P| = c_f(P)$.

Exemple dans quelques slides...

Le theoreme flot maximum / coupe minimum

Lemme (***)

Soit (S, T) une s, t -coupe de G et soit f un flot sur G . Alors, le flot net qui traverse (S, T) vaut $f(S, T) = |f|$.

Démonstration.

Soit $u \in S \setminus \{s\}$, montrez que $f(S \setminus \{u\}, T \cup \{u\}) = f(S, T)$.

Comment conclure ?

A-t-on besoin de montrer la meme chose dans le sens inverse,

c.a.d. que pour $v \in T \setminus \{t\}$, on a $f(S \cup \{v\}, T \setminus \{v\}) = f(S, T)$?



Le theoreme flot maximum / coupe minimum

Lemme (***)

Soit (S, T) une s, t -coupe de G et soit f un flot sur G . Alors, le flot net qui traverse (S, T) vaut $f(S, T) = |f|$.

Le theoreme flot maximum / coupe minimum

Lemme (***)

Soit (S, T) une s, t -coupe de G et soit f un flot sur G . Alors, le flot net qui traverse (S, T) vaut $f(S, T) = |f|$.

Démonstration.

Soit $u \in S \setminus \{s\}$, montrez que $f(S \setminus \{u\}, T \cup \{u\}) = f(S, T)$.

Comment conclure ?

A-t-on besoin de montrer la meme chose dans le sens inverse,

c.a.d. que pour $v \in T \setminus \{t\}$, on a $f(S \cup \{v\}, T \setminus \{v\}) = f(S, T)$?



Corollaire

Pour toute s, t -coupe (S, T) et tout flot f , on a $|f| \leq C(S, T)$.

Démonstration.

Par definition de $C(S, T)$ et de $f(S, T)$, on a directement $f(S, T) \leq C(S, T)$. Le lemme ci-dessus conclut.



Le theoreme flot maximum / coupe minimum

Théorème (***)

Soit G un reseau de flot et f un flot sur G . Les trois conditions suivantes sont equivalentes :

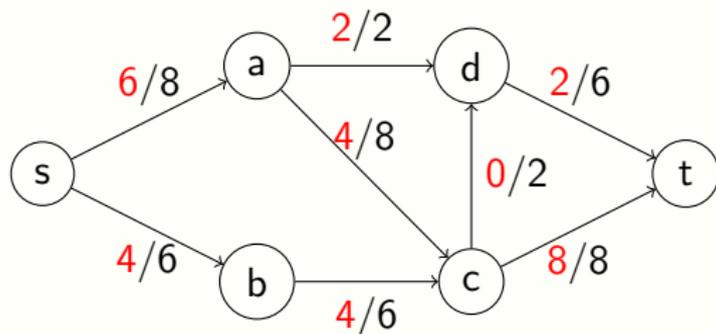
1. f est un flot maximum de G
2. le residuel G_f de f ne contient pas de chemin augmentant
3. \exists une s, t -coupe (S, T) telle que $|f| = C(S, T)$

Démonstration.

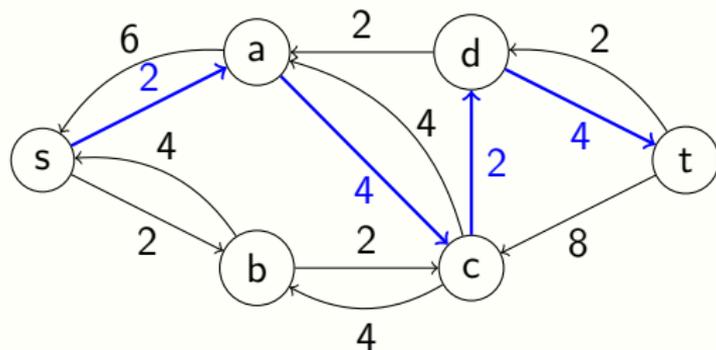
- **1** \Rightarrow **2**. D'apres la remarque sur les chemins augmentant on a $\bar{2} \Rightarrow \bar{1}$.
- **2** \Rightarrow **3**. Si 2 alors il existe une s, t -coupe (S, T) de capacite nulle dans G_f , ce qui implique $C_G(S, T) = |f|$ d'apres la remarque sur la capacite dans G_f des s, t -coupes.
- **3** \Rightarrow **1**. D'apres le corollaire precedent, on a $|f^*| \leq C(S, T)$, donc $|f| = |f^*|$.

Le theoreme flot maximum / coupe minimum

- $\bar{2} \Rightarrow \bar{1}$. ***

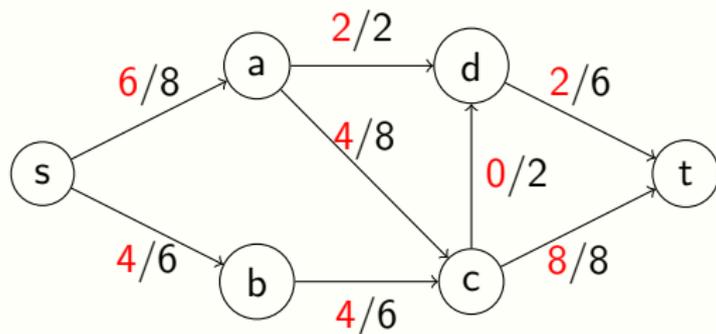


Le residuel G_f

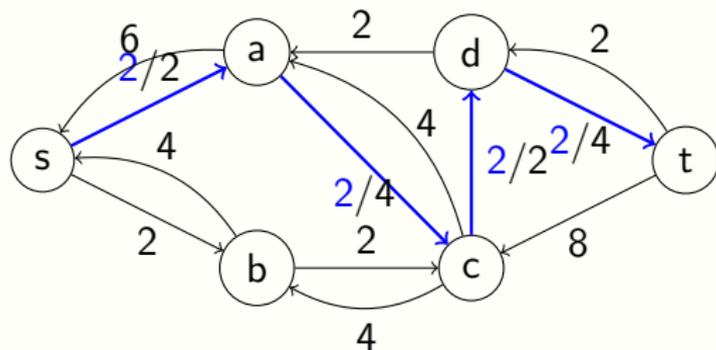


Le theoreme flot maximum / coupe minimum

- $\bar{2} \Rightarrow \bar{1}$. ***

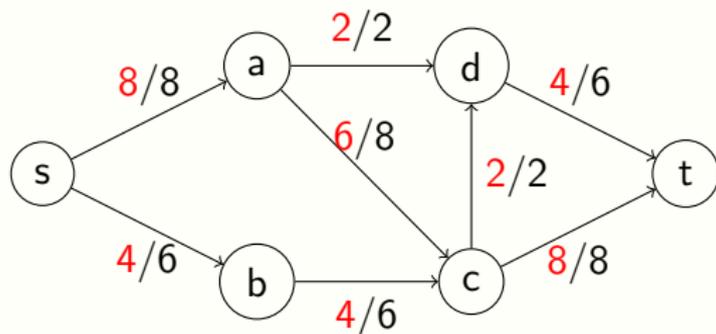


Le residuel G_f

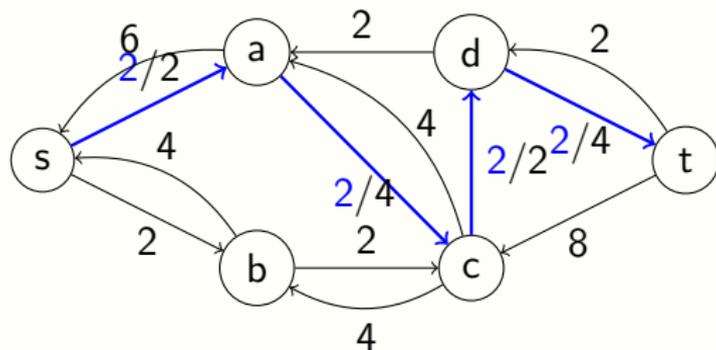


Le theoreme flot maximum / coupe minimum

- $\bar{2} \Rightarrow \bar{1}$. ***

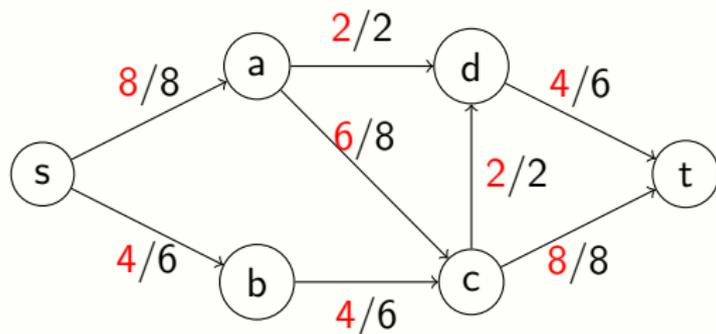


Le residuel G_f

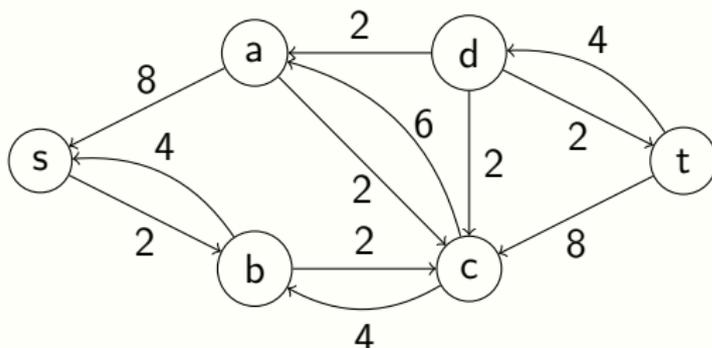


Le theoreme flot maximum / coupe minimum

- 2 \Rightarrow 3. ***

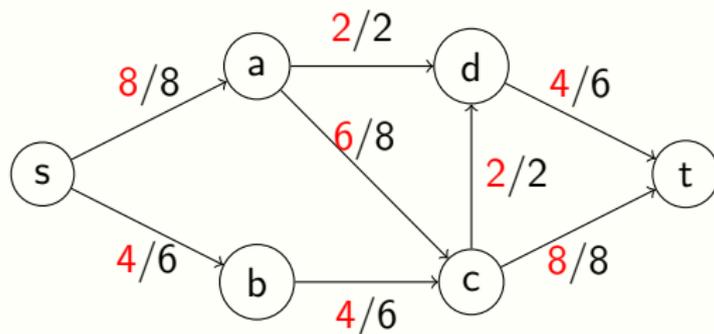


Le residuel G_f

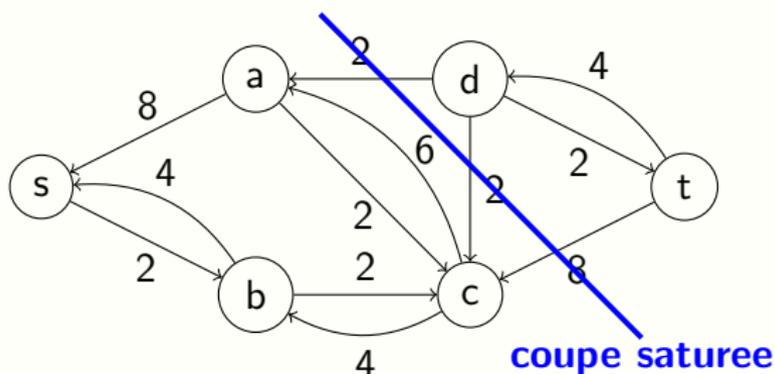


Le theoreme flot maximum / coupe minimum

- 2 \Rightarrow 3. ***



Le residuel G_f



Le theoreme flot maximum / coupe minimum

Question : A-t-on demontre le theoreme initial ?

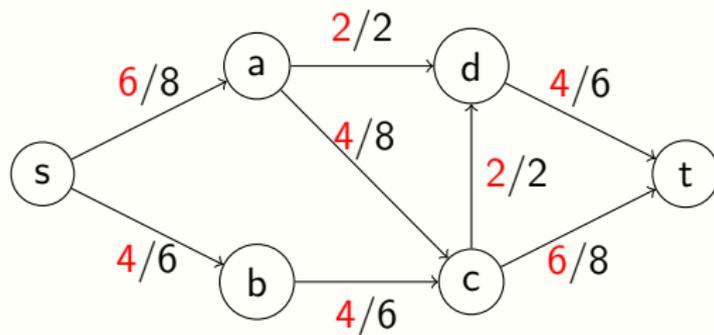
Le theoreme flot maximum / coupe minimum

Question : A-t-on demontre le theoreme initial ?

Pas tout a fait ! On a montre que s'il existe un maximum a la valeur des flots, alors c'est la capacite minimum d'une coupe, mais c'est pas sur qu'il existe bien un flot qui sature une coupe... On va en faire une demonstration algorithmique, qui a le merite de construire un flot maximum au passage.

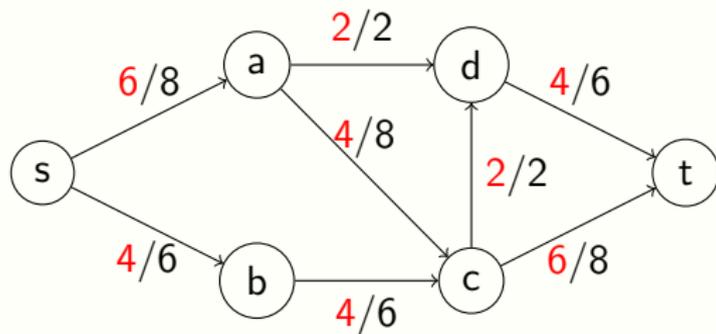
Rappels

Un flot f dans le reseau de flot G .



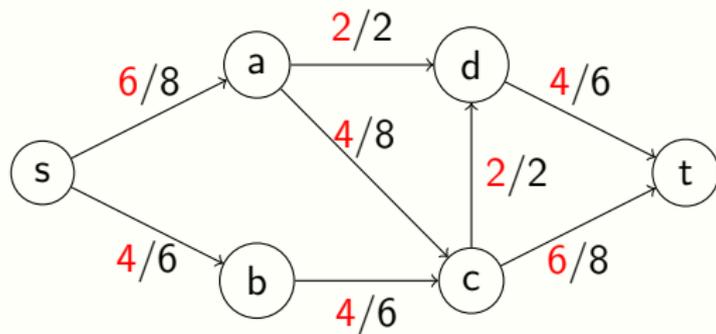
Rappels

Capacite d'une s, t -coupe et flot net a travers la coupe.



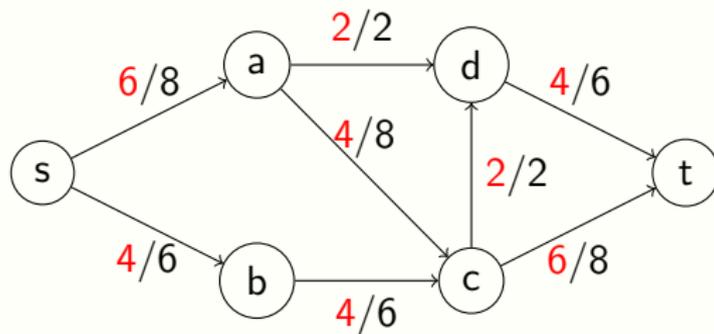
Rappels

Toutes les s, t -coupe ont le meme flot net !



Rappels

Question : flot net a travers une coupe qui ne separe pas s et t ?

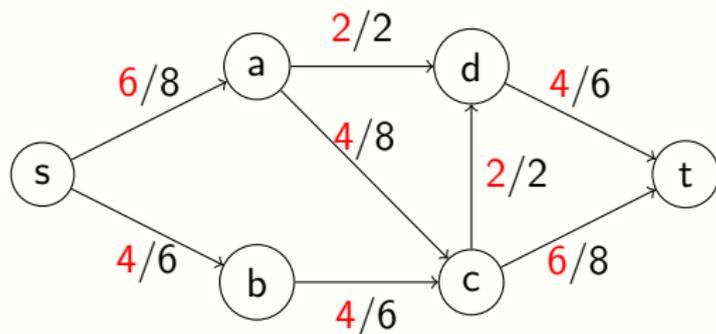


Rappels

Theoreme : flot maximum = coupe minimum

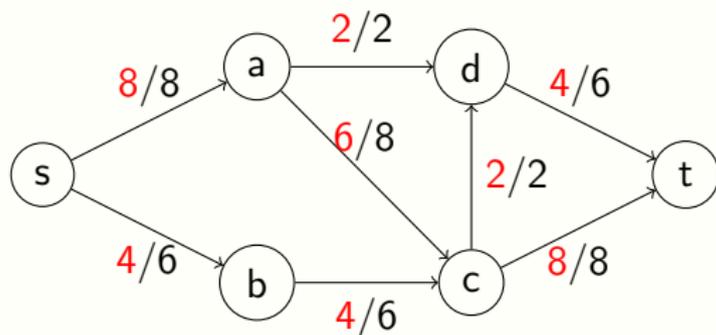
Reecriture du th (***) :

- le flot est maximum $\iff \exists$ une coupe saturee (= de capacite nulle dans le residuel)
- le flot n'est pas maximum $\iff \exists$ chemin augmentant dans le residuel

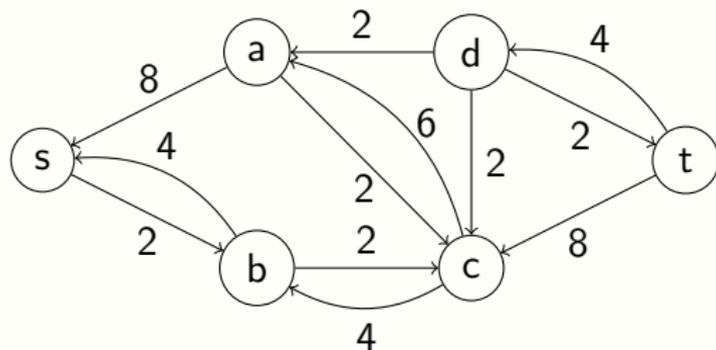


Rappels

- cas du flot maximum

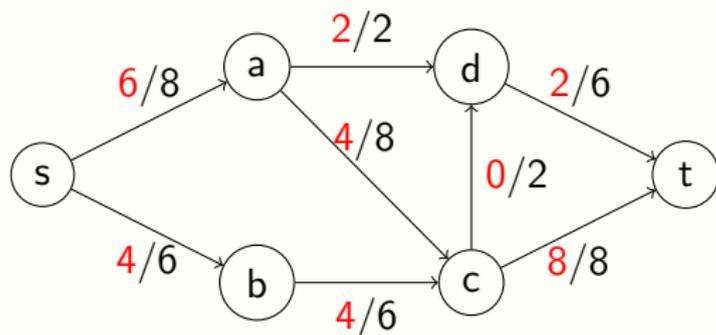


Le residuel G_f

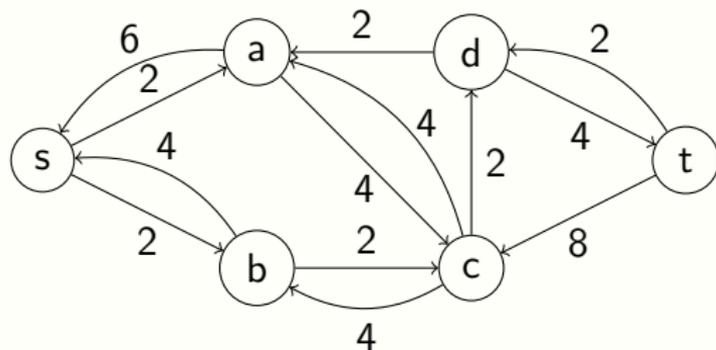


Rappels

- cas du flot NON maximum

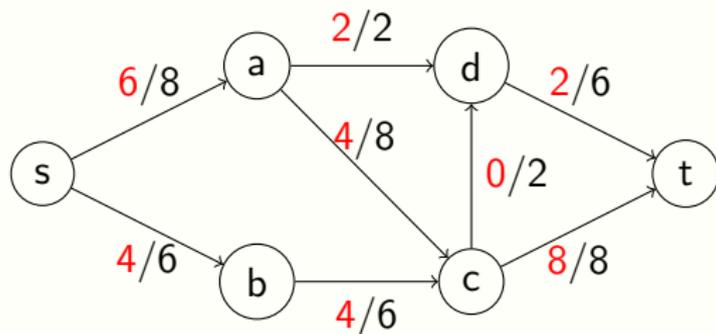


Le residuel G_f

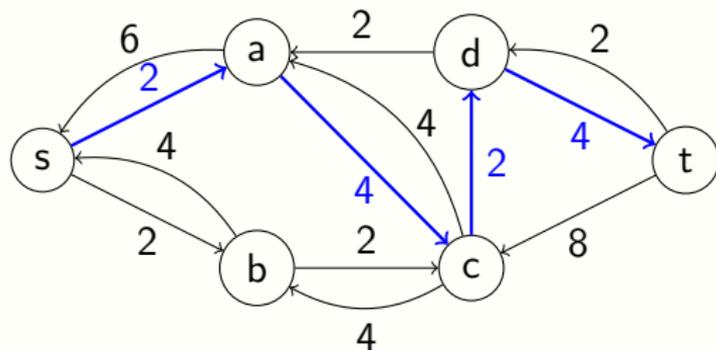


Rappels

- cas du flot NON maximum

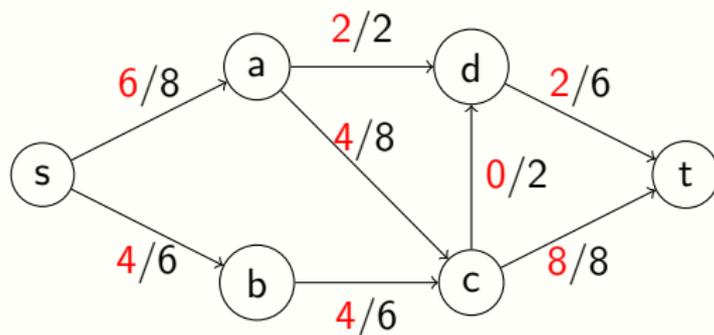


Le residuel G_f

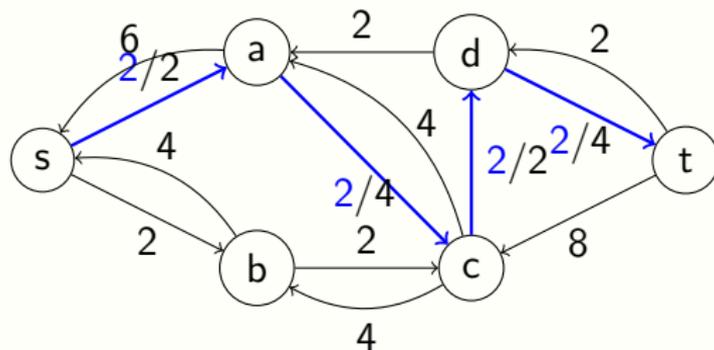


Rappels

- cas du flot NON maximum

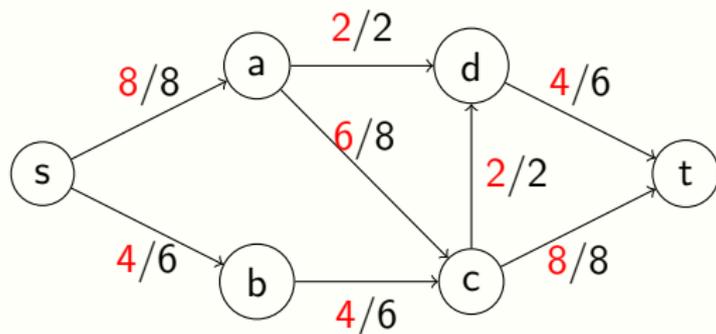


Le résiduel G_f

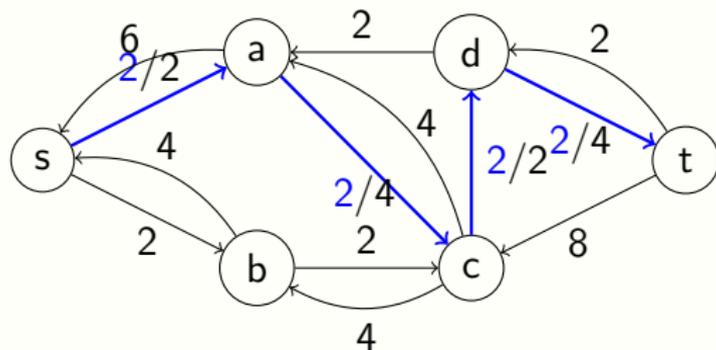


Rappels

- cas du flot NON maximum



Le residuel G_f



Rappels

- Existence de la coupe minimum
 - ▶ Il y a un nombre fini de s, t -coupes : exactement 2^{n-2} .

- Le probleme de l'existence du flot maximum
 - ▶ Il y a un nombre infini de flots dans G .
 - ▶ ...mais leur valeur est bornee.

Algorithme de Ford-Fulkerson (ne marche pas toujours)

Algorithme 1 : Ford-Fulkerson(G, s, t).

```
1 pour chaque arc  $(u, v) \in A$  faire  
2   |  $f(u, v) \leftarrow 0$ ;  
3 fin  
4 tant que  $\exists$  un chemin  $P$  de  $s$  a  $t$  dans le residuel  $G_f$  faire  
5   |  $c_f(P) \leftarrow \min\{c_f(u, v) \mid (u, v) \in P\}$ ;  
6   | pour chaque arc  $(u, v) \in P$  faire  
7     | si  $(u, v) \in A$  alors  $f(u, v) \leftarrow f(u, v) + c_f(P)$ ;  
8     | sinon  $f(v, u) \leftarrow f(v, u) - c_f(P)$ ;  
9   | fin  
10 fin  
11 retourner  $f$ ;
```

Domaine de validite de Ford-Fulkerson

- Capacites entieres

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**
 - ▶ Terminaison : OK

Domaine de validite de Ford-Fulkerson

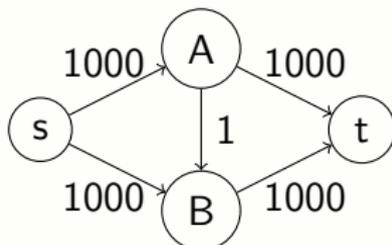
- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

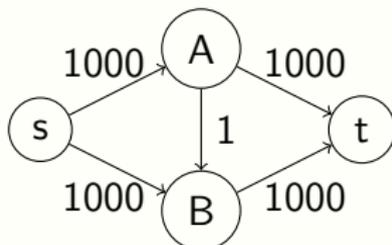
- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$



Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$

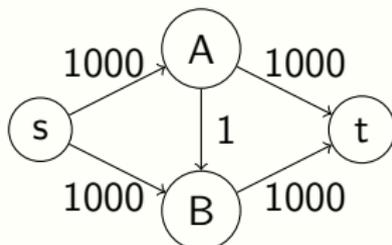


- **Capacites rationnelles**

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$



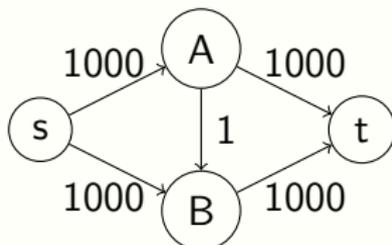
- **Capacites rationnelles**

- ▶ Terminaison : se ramener aux capacites entieres en multipliant par le ppcm des denominateurs $c(u, v) \rightarrow \tilde{c}(u, v)$

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$



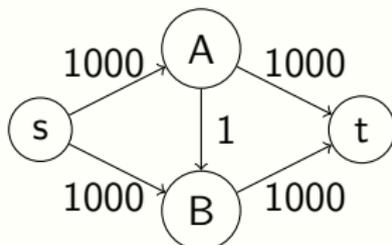
- **Capacites rationnelles**

- ▶ Terminaison : se ramener aux capacites entieres en multipliant par le ppcm des denominateurs $c(u, v) \rightarrow \tilde{c}(u, v)$
- ▶ Complexite : $O(|\tilde{f}^*|m)$

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$



- **Capacites rationnelles**

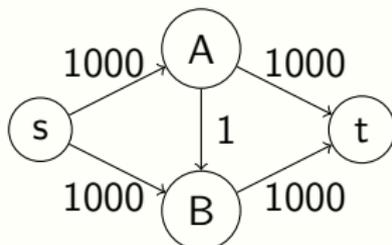
- ▶ Terminaison : se ramener aux capacites entieres en multipliant par le ppcm des denominateurs $c(u, v) \rightarrow \tilde{c}(u, v)$
- ▶ Complexite : $O(|\tilde{f}^*|m)$

- **Capacites reelles (notre cadre d'etude initial)**

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$



- **Capacites rationnelles**

- ▶ Terminaison : se ramener aux capacites entieres en multipliant par le ppcm des denominateurs $c(u, v) \rightarrow \tilde{c}(u, v)$
- ▶ Complexite : $O(|\tilde{f}^*|m)$

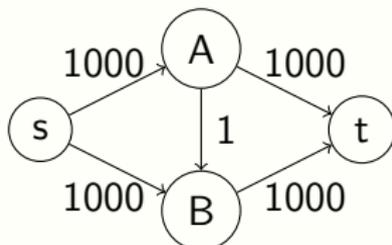
- **Capacites reelles (notre cadre d'etude initial)**

- ▶ Terminaison : NON (voir exemple wikipedia avec capacites irrationnelles)

Domaine de validite de Ford-Fulkerson

- **Capacites entieres**

- ▶ Terminaison : OK
- ▶ Complexite : $O(|f^*|m)$, avec $m = |A|$



- **Capacites rationnelles**

- ▶ Terminaison : se ramener aux capacites entieres en multipliant par le ppcm des denominateurs $c(u, v) \rightarrow \tilde{c}(u, v)$
- ▶ Complexite : $O(|\tilde{f}^*|m)$

- **Capacites reelles (notre cadre d'etude initial)**

- ▶ Terminaison : NON (voir exemple wikipedia avec capacites irrationnelles)
- ▶ Convergence : pas vers $|f^*| \dots$ et meme aussi loin "qu'on veut" de $|f^*|$ (exemple wikipedia)

Capacités réelles : Ford-Fulkerson \rightarrow Edmonds-Karp

- Ligne 4 de FF : un chemin $P \rightarrow$ un plus court chemin P (en nombre d'arcs)... c'est tout !

Capacites reelles : Ford-Fulkerson \rightarrow Edmonds-Karp

- Ligne 4 de FF : un chemin $P \rightarrow$ un plus court chemin P (en nombre d'arcs)... c'est tout !
- Terminaison : garantie meme pour capacites irrationnelles

Capacités réelles : Ford-Fulkerson \rightarrow Edmonds-Karp

- Ligne 4 de FF : un chemin $P \rightarrow$ un plus court chemin P (en nombre d'arcs)... c'est tout !
- Terminaison : garantie même pour capacités irrationnelles
- Complexité : $O(nm^2)$, avec $n = |V|$ et $m = |A|$

Validite et complexite de l'algorithme d'Edmonds-Karp

Lemme

Après chaque augmentation du flot dans l'algo d'EK, pour tout sommet $v \in V$, la distance (en nombre d'arcs) de s à v dans le résiduel ne décroît pas.

Validite et complexite de l'algorithme d'Edmonds-Karp

Lemme

Après chaque augmentation du flot dans l'algo d'EK, pour tout sommet $v \in V$, la distance (en nombre d'arcs) de s à v dans le résiduel ne décroît pas.

Démonstration.

f et f' le flot avant et après augmentation.

Par l'absurde : supposons $\exists v \in V, \delta_{G_f}(s, v) > \delta_{G_{f'}}(s, v)$.

Soit v un tel sommet dont la distance à s dans $G_{f'}$ est minimum.

Soit u le précédent de v sur un plus court chemin de s à v dans $G_{f'}$.

Proposition

$(u, v) \notin E_f$

Par conséquent, le flot a été augmenté sur l'arc (v, u) . On montre alors que $\delta_{G_f}(s, v) \leq \delta_{G_{f'}}(s, v) - 2$: contradiction. \square

Validite et complexite de l'algorithme d'Edmonds-Karp

Définition

Une arete du residuel est *critique* si elle se trouve sur le chemin P choisit par l'algorithme d'EK et que sa capacite dans G_f vaut precisement $c_f(P)$.

Lemme

Une arete ne peut devenir critique qu'au plus $n/2$ fois au cours de l'algorithme d'EK.

Validite et complexite de l'algorithme d'Edmonds-Karp

Définition

Une arete du residuel est *critique* si elle se trouve sur le chemin P choisit par l'algorithme d'EK et que sa capacite dans G_f vaut precisement $c_f(P)$.

Lemme

Une arete ne peut devenir critique qu'au plus $n/2$ fois au cours de l'algorithme d'EK.

Démonstration.

Soit f le flot lorsque (u, v) est critique et f' le flot la prochaine fois que (v, u) est sur le chemin choisit par EK.

On peut montrer que $\delta_{G_{f'}}(s, u) \geq \delta_{G_f}(s, u) + 2$. Et comme $\delta(s, u)$ ne peut excéder $n - 2$, (u, v) ne peut devenir critique qu'au plus $n/2$ fois. □

Validite et complexite de l'algorithme d'Edmonds-Karp

Corollaire

La complexite de l'algorithme d'EK est $O(nm^2)$.

Validite et complexite de l'algorithme d'Edmonds-Karp

Corollaire

La complexite de l'algorithme d'EK est $O(nm^2)$.

Démonstration.

D'apres le lemme precedent, le nombre de chemin augmentant dans l'algo d'EK ne peut excéder $mn/2$.

Trouver un plus court chemin augmentant prend un temps $O(n + m)$, par un BFS par exemple.

⇒ Complexite totale de $O(nm^2)$ pour l'algo d'EK. □

Validite et complexite de l'algorithme d'Edmonds-Karp

Corollaire

La complexite de l'algorithme d'EK est $O(nm^2)$.

Démonstration.

D'apres le lemme precedent, le nombre de chemin augmentant dans l'algo d'EK ne peut excéder $mn/2$.

Trouver un plus court chemin augmentant prend un temps $O(n + m)$, par un BFS par exemple.

⇒ Complexite totale de $O(nm^2)$ pour l'algo d'EK. □

- **L'algorithme d'EK termine !** (meme avec capacites irrationnelles)

Validite et complexite de l'algorithme d'Edmonds-Karp

Corollaire

La complexite de l'algorithme d'EK est $O(nm^2)$.

Démonstration.

D'apres le lemme precedent, le nombre de chemin augmentant dans l'algo d'EK ne peut excéder $mn/2$.

Trouver un plus court chemin augmentant prend un temps $O(n + m)$, par un BFS par exemple.

⇒ Complexite totale de $O(nm^2)$ pour l'algo d'EK. □

- **L'algorithme d'EK termine !** (meme avec capacites irrationnelles)

Et comme il termine sur un flot f dont le reseau residuel n'a pas de chemin augmentant, d'apres le theoreme principal, f est maximum.

Validite et complexite de l'algorithme d'Edmonds-Karp

Corollaire

La complexite de l'algorithme d'EK est $O(nm^2)$.

Démonstration.

D'apres le lemme precedent, le nombre de chemin augmentant dans l'algo d'EK ne peut excéder $mn/2$.

Trouver un plus court chemin augmentant prend un temps $O(n + m)$, par un BFS par exemple.

⇒ Complexite totale de $O(nm^2)$ pour l'algo d'EK. □

- **L'algorithme d'EK termine !** (meme avec capacites irrationnelles)

Et comme il termine sur un flot f dont le reseau residuel n'a pas de chemin augmentant, d'apres le theoreme principal, f est maximum.

- **L'algorithme d'EK est correct !**
... et le flot maximum existe toujours ! ;)

Tableau Blanc

Tableau Blanc