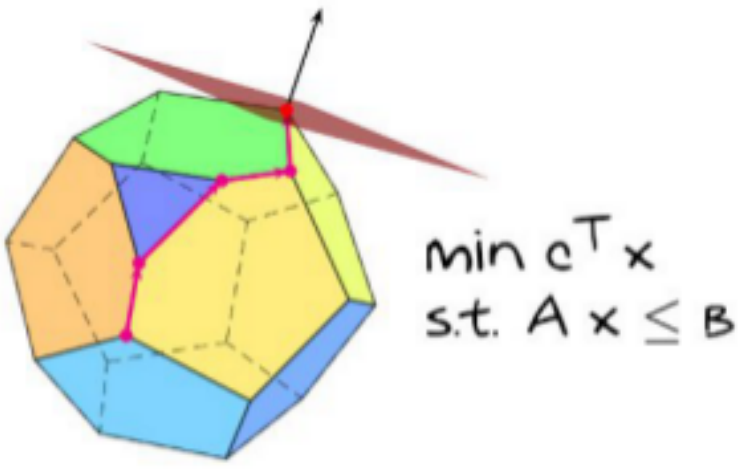
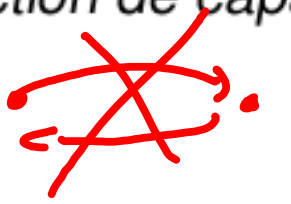
Flots

- Réseaux et Flots



Réseaux et flots

Réseau: Graphe orienté simple $D = (V, A)$ et *fonction de capacité* $u : A \to \mathbb{R}_{\geqslant 0}$.

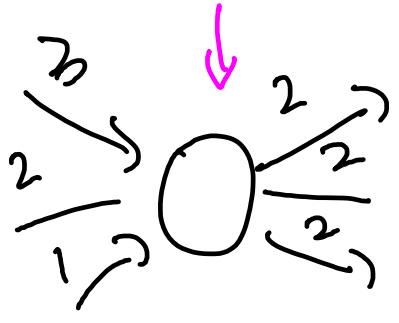


$f : A \to \mathbb{R}_{\geqslant 0}$ est appelée *flot de s à t*, si

$$\sum_{e \in \delta^{out}(v)} f(e) = \sum_{e \in \delta^{in}(v)} f(e), \text{ pour tout } v \in V - \{s, t\}, \qquad (1)$$

où $s, t \in V$. Le flot est *admissible*, si $f(e) \leqslant u(e)$ pour tout $e \in A$.

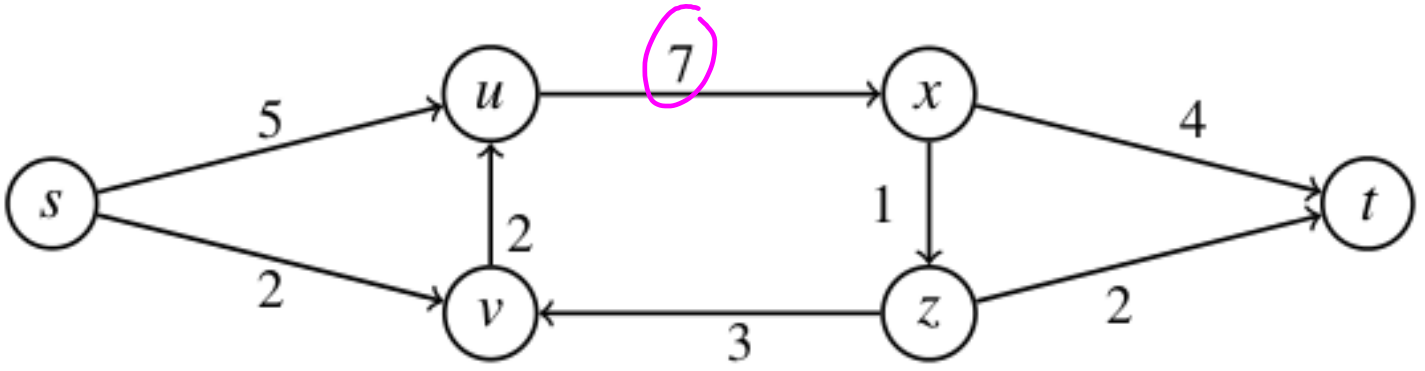
flot ⇒ (s)



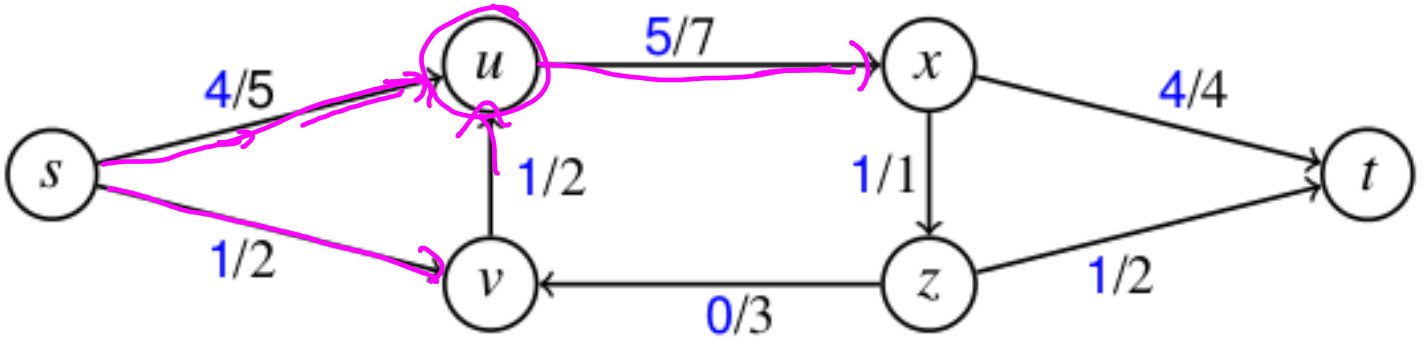
(t) ⇒

Exemple

capacités



Exemple : Réseau et flot de s à t

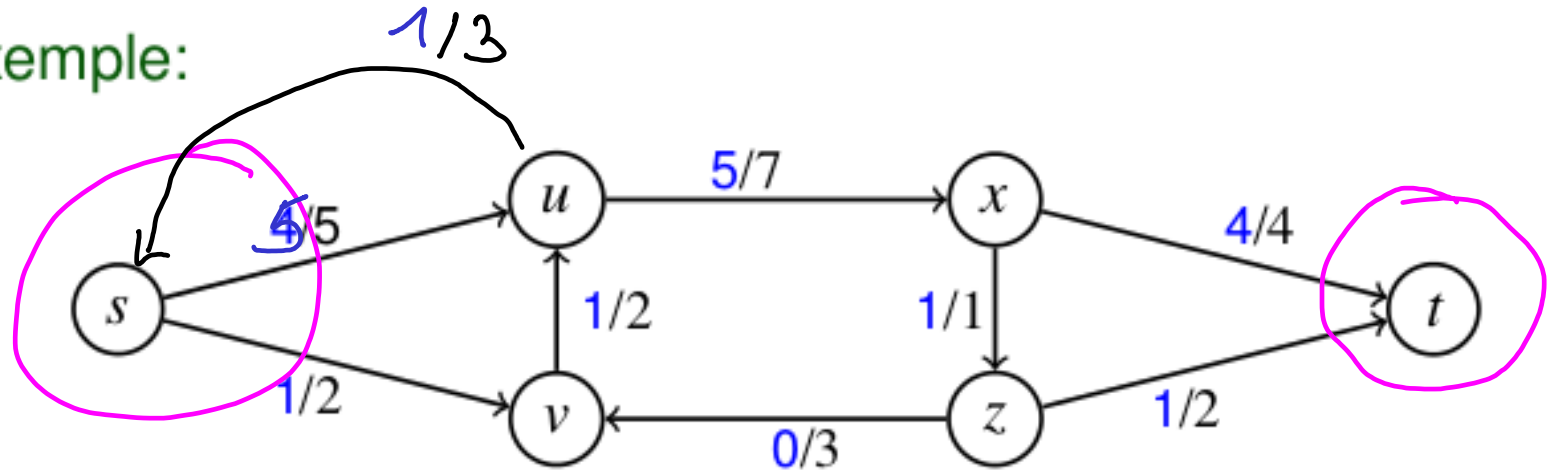


Valeur du flot

La *valeur* de f est définie comme
$value(f) = \sum_{e \in \delta^{out}(s)} f(e) - \sum_{e \in \delta^{in}(s)} f(e)$.

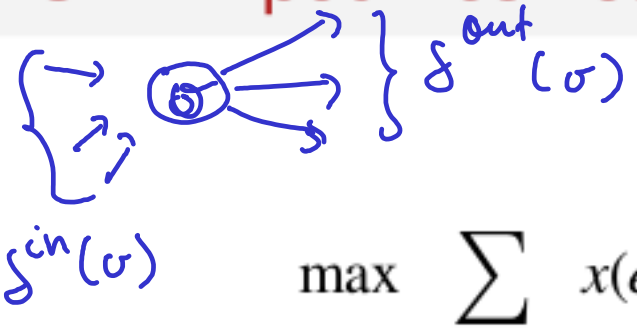
Le *problème du flot maximal de s à t* consiste à déterminer un flot valeur maximal de s à t qui est admissible.

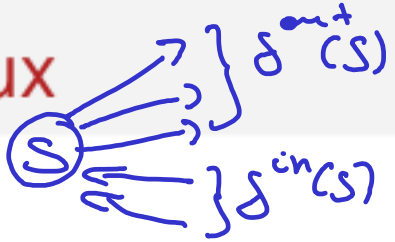
Exemple:

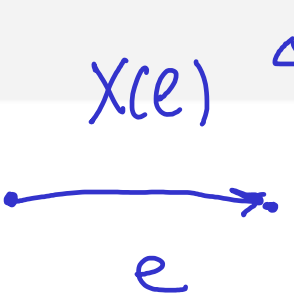


Valeur = 5

Un PL pour les flots maximaux







$$\max \sum_{e \in \delta^{out}(s)} x(e) - \sum_{e \in \delta^{in}(s)} x(e)$$

$$\sum_{e \in \delta^{out}(v)} x(e) = \sum_{e \in \delta^{in}(v)} x(e), \text{ pour tout } v \in V - \{s,t\}$$

$$x(e) \leqslant u(e), \text{ pour tout } e \in A$$

Capacités sont respectés

$$x(e) \geqslant 0, \text{ pour tout } e \in A$$

Remarque

Si le PL est borné et les capacités sont des nombres entiers, le PL a une solution optimale intégrale, lorsque la *matrice d'incidence sommets-arcs d'un graphe orienté* est totalement unimodulaire.

Un PL pour les flots maximaux

$$\max \sum_{e \in \delta^{out}(s)} x(e) - \sum_{e \in \delta^{in}(s)} x(e)$$

$$\sum_{e \in \delta^{out}(v)} x(e) = \sum_{e \in \delta^{in}(v)} x(e), \text{ pour tout } v \in V - \{s, t\}$$

$$x(e) \leqslant u(e), \text{ pour tout } e \in A$$

$$x(e) \geqslant 0, \text{ pour tout } e \in A$$

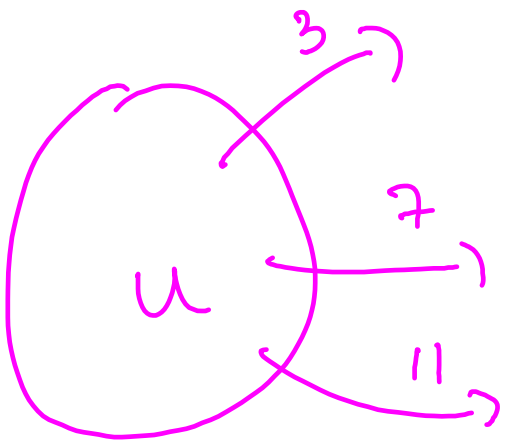
Remarque

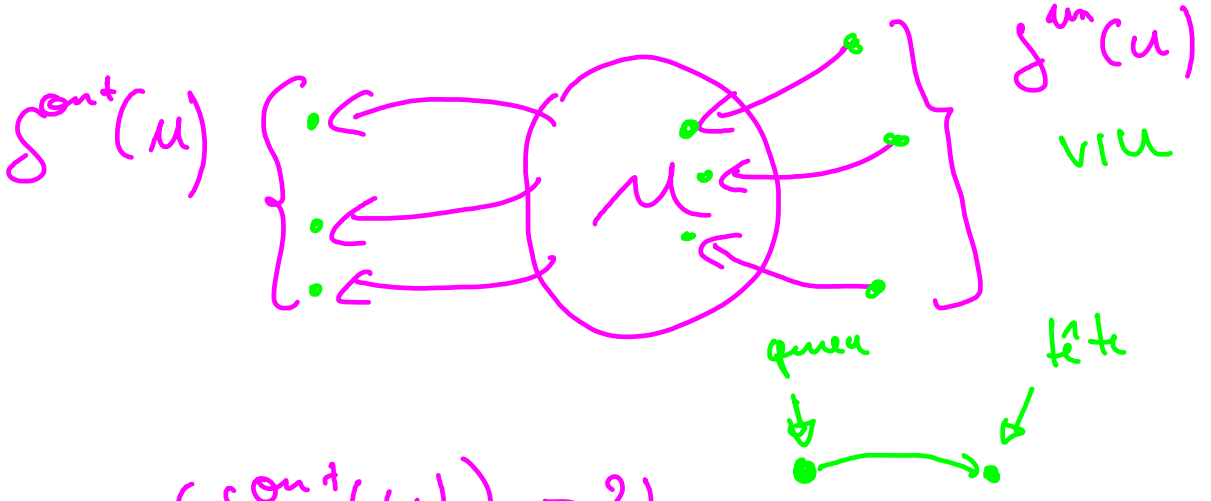
Si le PL est borné et les capacités sont des nombres entiers, le PL a une solution optimale intégrale, lorsque la *matrice d'incidence sommets-arcs d'un graphe orienté* est totalement unimodulaire.

Coupes

Pour $U \subseteq V$, $\delta^{in}(U)$ dénote les arcs qui entrent dans U et $\delta^{out}(U)$ dénote les arcs sortant de U.
Des ensembles d'arcs de la forme $\delta^{out}(U)$ sont appelés une *coupe* de D.

La *capacité d'une coupe* $u(\delta^{out}(U))$ est la somme des capacités de ses arcs.





Coupes

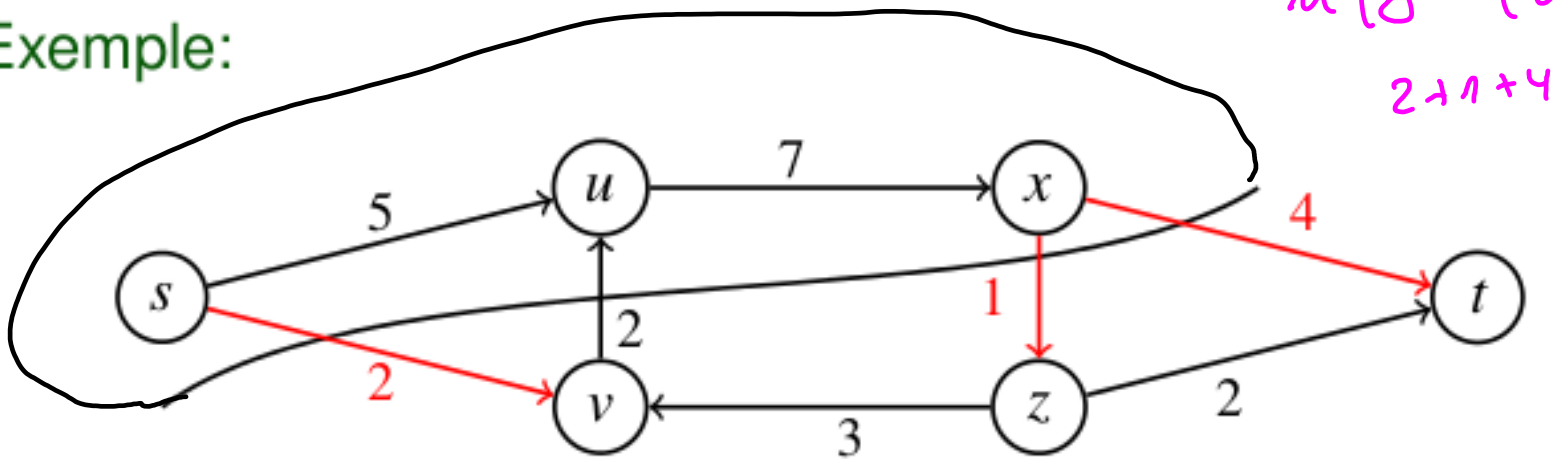
Pour $U \subseteq V$, $\delta^{in}(U)$ dénote les arcs qui entrent dans U et $\delta^{out}(U)$ dénote les arcs sortant de U.
Des ensembles d'arcs de la forme $\delta^{out}(U)$ sont appelés une *coupe* de D.

si $s \in U$ et $t \notin U$, $\delta^{out}(U)$ est une coupe s-t

La *capacité d'une coupe* $u(\delta^{out}(U))$ est la somme des capacités de ses arcs.

$u(\delta^{out}(U)) =$
$2+1+4 = 7$

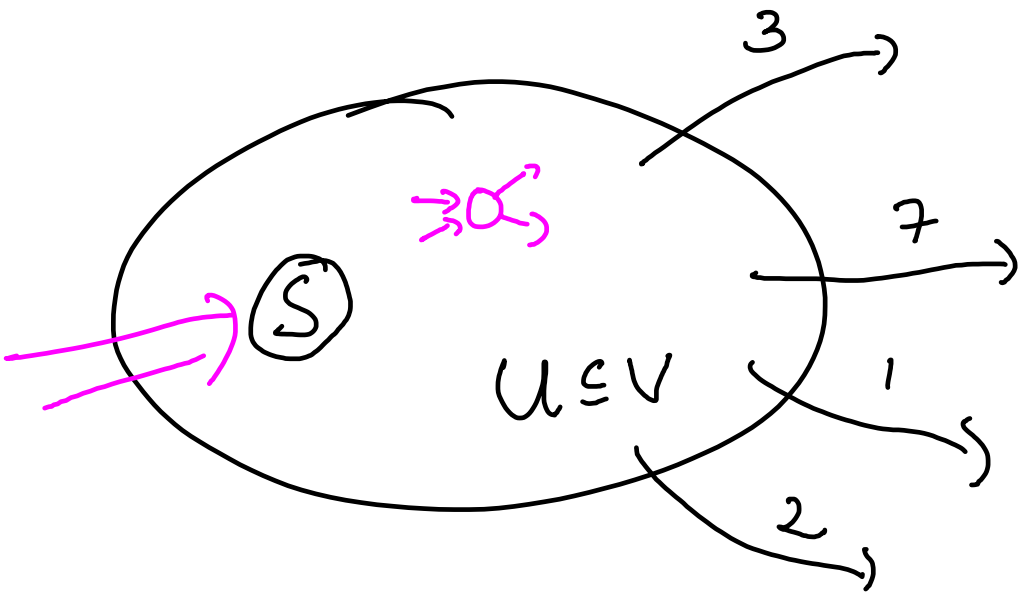
Exemple:



Fonction d'excès

Pour tout $f : A \to \mathbb{R}$, *la fonction d'excès*; $excès_f : 2^V \to \mathbb{R}$ est définie par

$$excès_f(U) = \sum_{e \in \delta^{in}(U)} f(e) - \sum_{e \in \delta^{out}(U)} f(e).$$



t

Idée : Flot

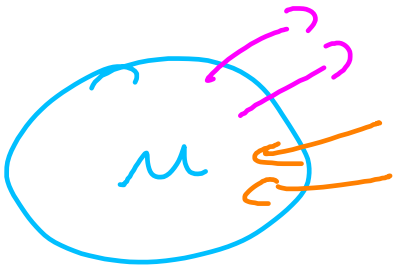
max. est ≤ 13

But : Confirmer notre intuition.

Fonction d'excès

Pour tout $f : A \to \mathbb{R}$, *la fonction d'excès*; $excès_f : 2^V \to \mathbb{R}$ est définie par

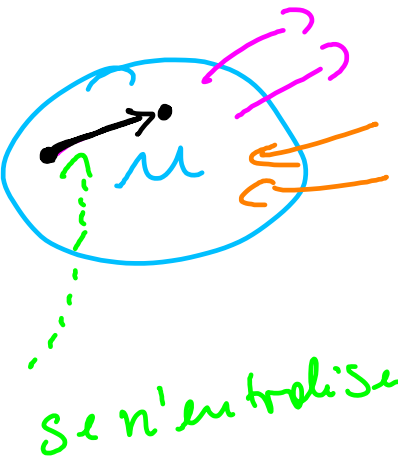
$$excès_f(U) = \sum_{e \in \delta^{in}(U)} f(e) - \sum_{e \in \delta^{out}(U)} f(e).$$



Fonction d'excès

Soit $D = (V, A)$ un graphe orienté, soit $f : A \to \mathbb{R}$ et soit $U \subseteq V$, alors

$$excès_f(U) = \sum_{v \in U} excès_f(v). = \sum_{e \in \delta^{in}(U)} f(e) - \sum_{e \in \delta^{out}(U)} f(e)$$



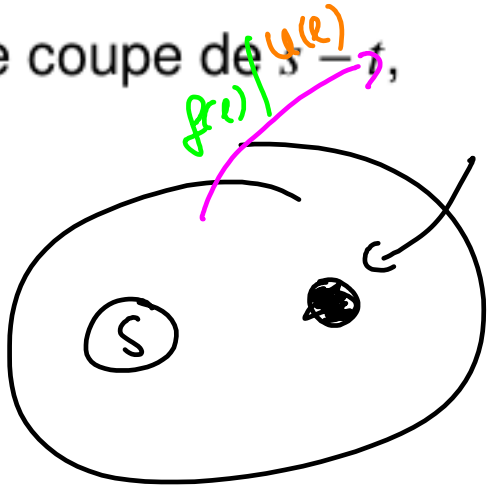
$= excès_f(U)$

Dualité faible

Dualité faible

Soit f un flot admissible de s à t et soit $\delta^{out}(U)$ une coupe de $s-t$, alors $value(f) \leqslant u(\delta^{out}(U))$.

excès =



$$value(f) = -excès(s)$$

$$= \sum_{v \in U} -excès(v)$$

$$= -excès(U) = \sum_{e \in \delta^{out}(U)} f(e) - \sum_{e \in \delta^{in}(U)} f(e)$$

$$\leq \sum_{e \in \delta^{out}(U)} f(e) \leq u(\delta^{out}(U))$$

$f(e) \leq u(e)$

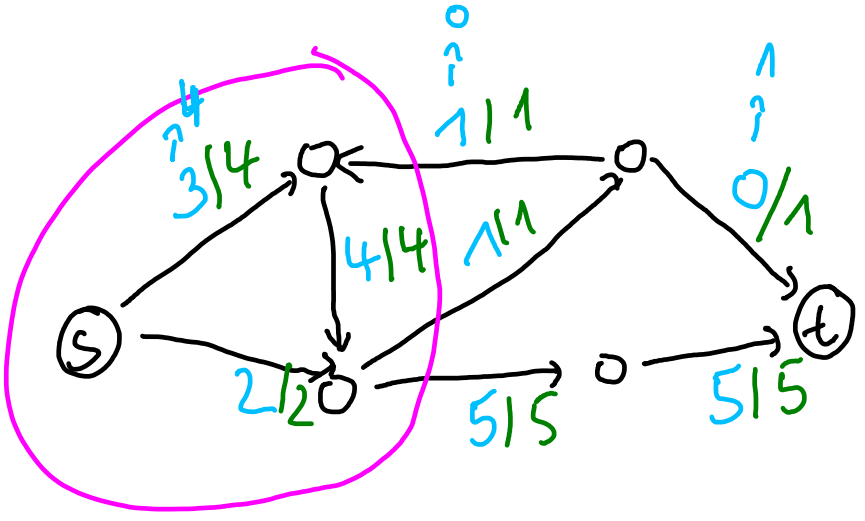
∎

Le Graphe résiduel

Soit $f : A \to \mathbb{R}$ et $u : A \to \mathbb{R}$ où $0 \leqslant f \leqslant u$. Considérons l'ensemble d'arcs

$$A_f = \{a \mid a \in A, f(a) < u(a)\} \cup \{a^{-1} \mid a \in A, f(a) > 0\}.$$

Le graphe orienté $D(f) = (V, A_f)$ est appelé *graphe résiduel* de f (pour des capacités u).

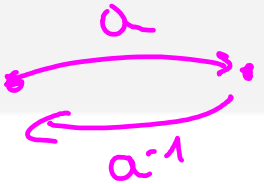


Capacité = 6

nouvelle valeur = 6

= 1 opt !

Le Graphe résiduel

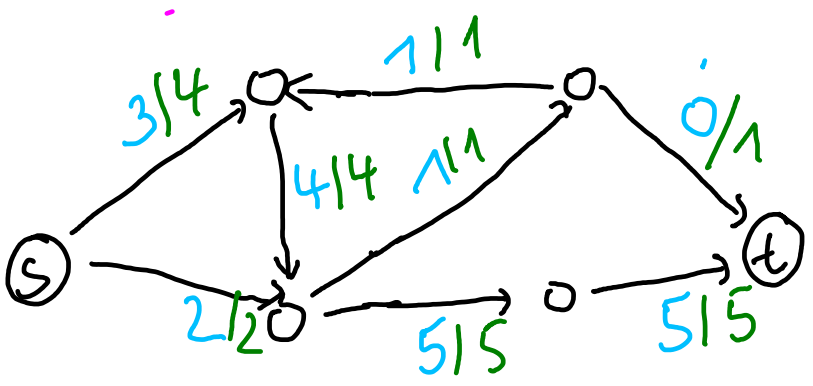


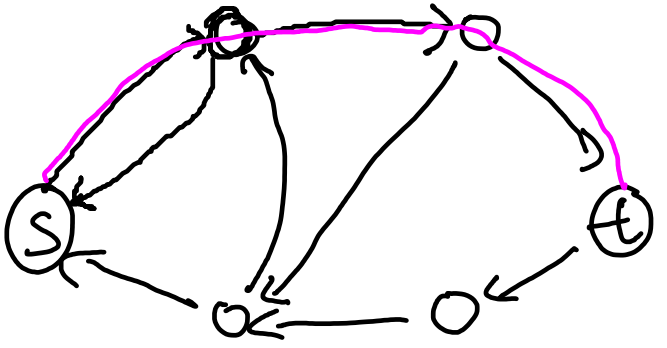
Soit $f : A \to \mathbb{R}$ et $u : A \to \mathbb{R}$ où $0 \leqslant f \leqslant u$. Considérons l'ensemble d'arcs

$$A_f = \{a \mid a \in A, f(a) < u(a)\} \cup \{a^{-1} \mid a \in A, f(a) > 0\}.$$

Le graphe orienté $D(f) = (V, A_f)$ est appelé *graphe résiduel* de f (pour des capacités u).

Chemin orienté de s à t !!





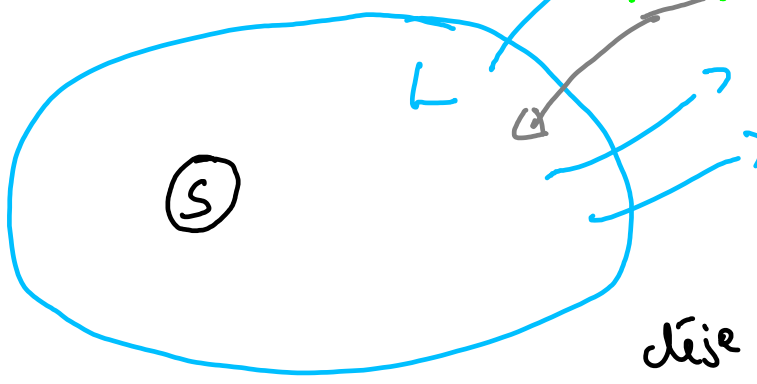
Graphe résiduel !!

Théorème

Soit f un flot admissible de s à t et supposons qu'il n'y a pas de chemin de s à t dans $D(f)$, alors f est de valeur maximale.

Démonstration:

Considerons $D(f)$:



$L \subseteq V$ $L = d^-$: ∃ chemin de s à v dans $D(f)$

$t \notin L$

déjà démontré

$$u\left(\delta^{out}(L)\right) \overset{2.}{=} -\text{exces}_f(L) \overset{1.}{=} -\text{exces}(s) = \text{valeur de } f.$$

∎

Marches non-orientées

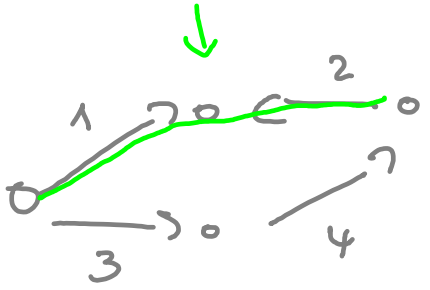
Une *marche non-orientée* est une séquence de la forme $P = (v_0, v_1, \ldots, v_{m-1}, v_m)$, où $(v_{i-1}, v_i) \in A$ ou $(v_i, v_{i-1}) \in A$. Si les sommets $v_0, \ldots, v_m$ sont tous différents, alors P est un *chemin non-orienté*.

$D(f)$ et marches non-orientées

Tout chemin orienté P dans $D(f)$ induit un chemin non-orienté dans D. Définissons pour un tel chemin P le vecteur $\chi^P \in \{0, \pm 1\}^A$ comme

$$\chi^P(a) = \begin{cases} 1 & \text{si } P \text{ traverse } a, \\ -1 & \text{si } P \text{ traverse } a^{-1}, \\ 0 & \text{si } P \text{ ne traverse ni } a \text{ ni } a^{-1}. \end{cases} \tag{2}$$

chemin non-orienté.



$$\chi^P(a) = \begin{pmatrix} 1 \\ -1 \\ 0 \\ 0 \end{pmatrix}$$

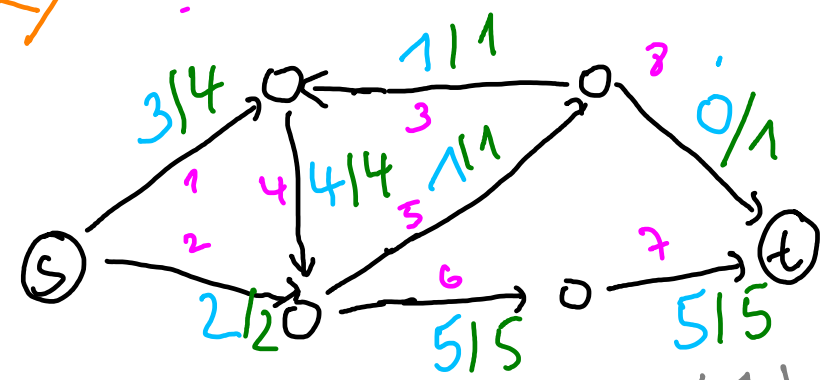
$D(f)$ et marches non-orientées

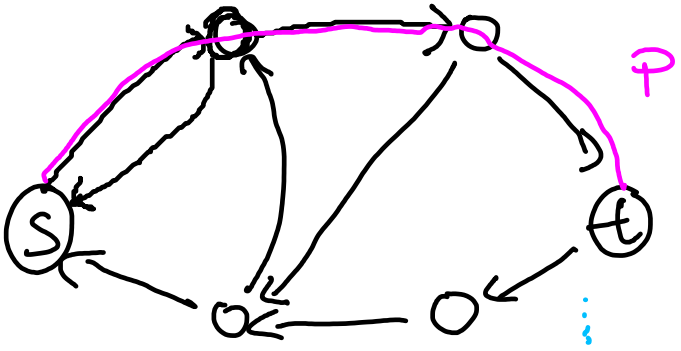
Tout chemin orienté P dans $D(f)$ induit un chemin non-orienté dans D. Définissons pour un tel chemin P le vecteur $\chi^P \in \{0, \pm 1\}^A$ comme

$$\chi^P(a) = \begin{cases} 1 & \text{si } P \text{ traverse } a, \\ -1 & \text{si } P \text{ traverse } a^{-1}, \\ 0 & \text{si } P \text{ ne traverse ni } a \text{ ni } a^{-1} \end{cases} \qquad (2)$$

nouveau flot $f + \chi^P$!!

Chemin orienté de s à t !!





χ^P : $\begin{pmatrix} 1 \\ 0 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$ $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$

$f = \begin{pmatrix} 3 \\ 2 \\ 1 \\ 4 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$ $\begin{pmatrix} 5 \\ 5 \\ 0 \end{pmatrix}$

L'algorithme de Ford et Fulkerson

Commencer avec $f = 0$. Puis appliquer itérativement *l'algorithme d'augmentation de flot* suivant:

Soit P un chemin orienté de s à t dans $D(f)$. Soit $f \leftarrow f + \epsilon\chi^P$, où ϵ est aussi grand que possible tout en gardant $0 \leqslant f \leqslant u$.

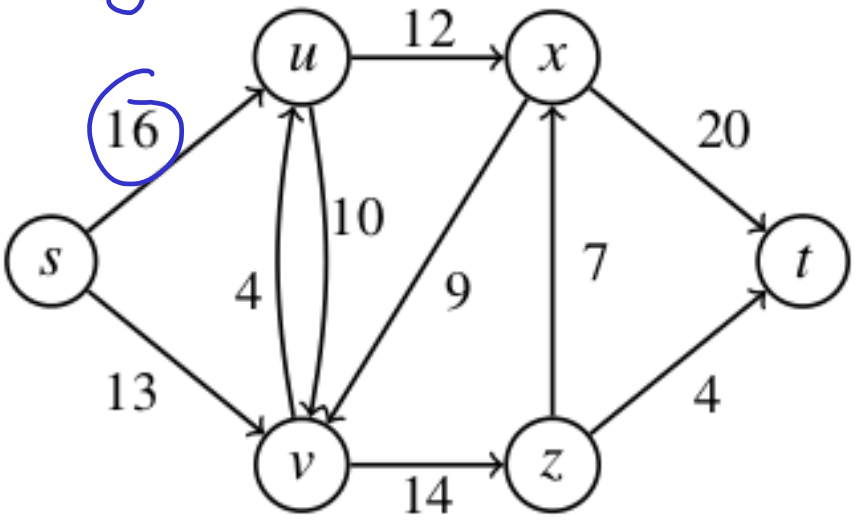
Quiz

Définissez une *capacité résiduelle* pour $D(f)$. Déterminez ensuite la valeur maximale de ϵ telle que $0 \leqslant f \leqslant u$.

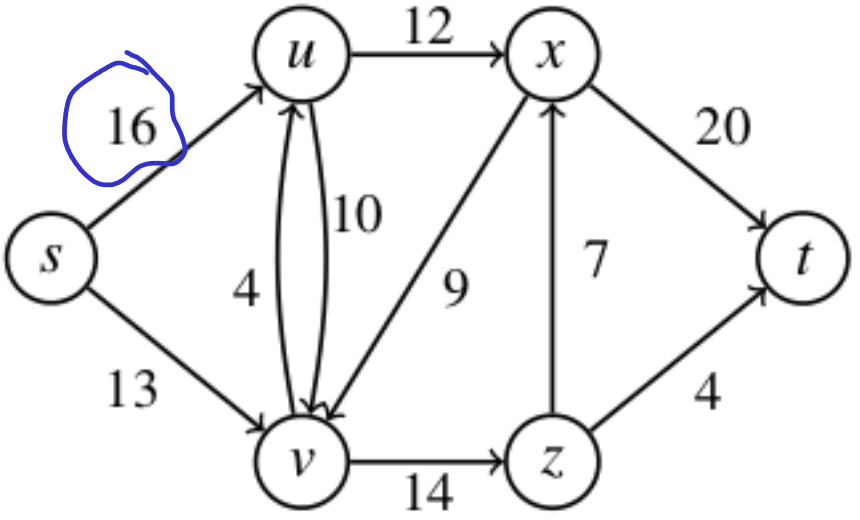
Supposons: D n'a pas

Algorithme de Ford-Fulkerson : exemple

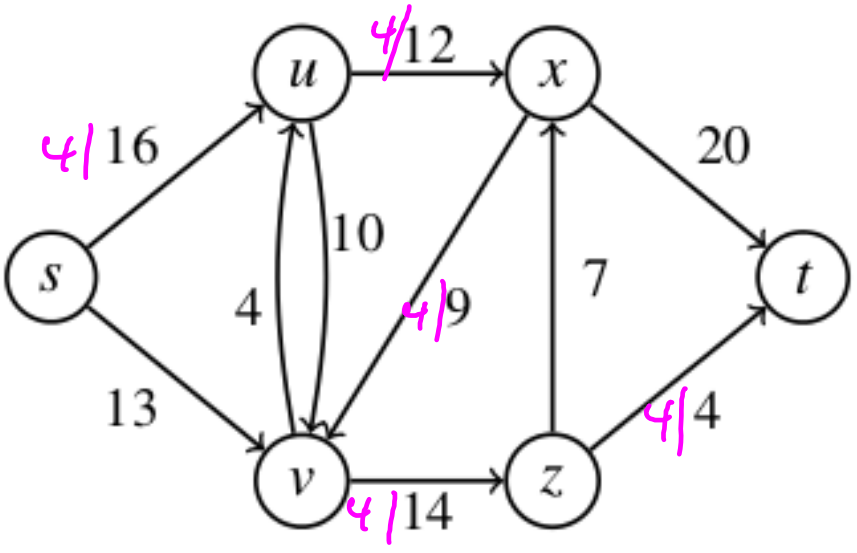
capacités.

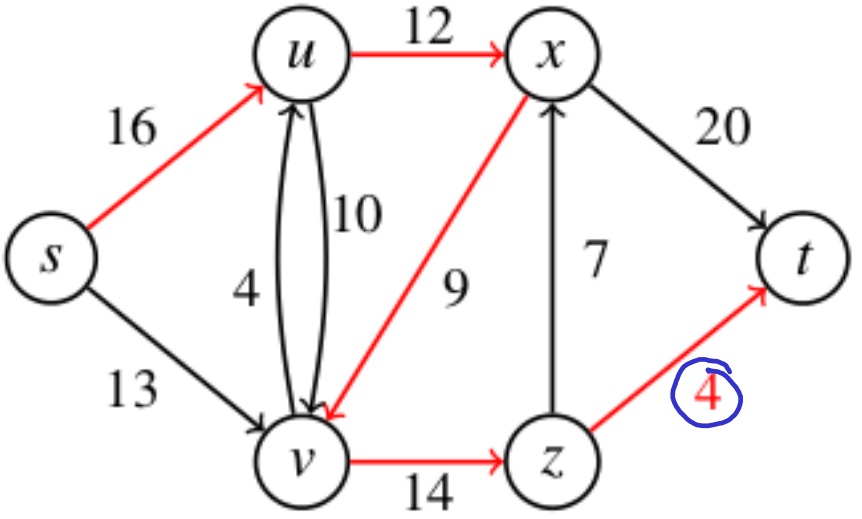


cap. rés.

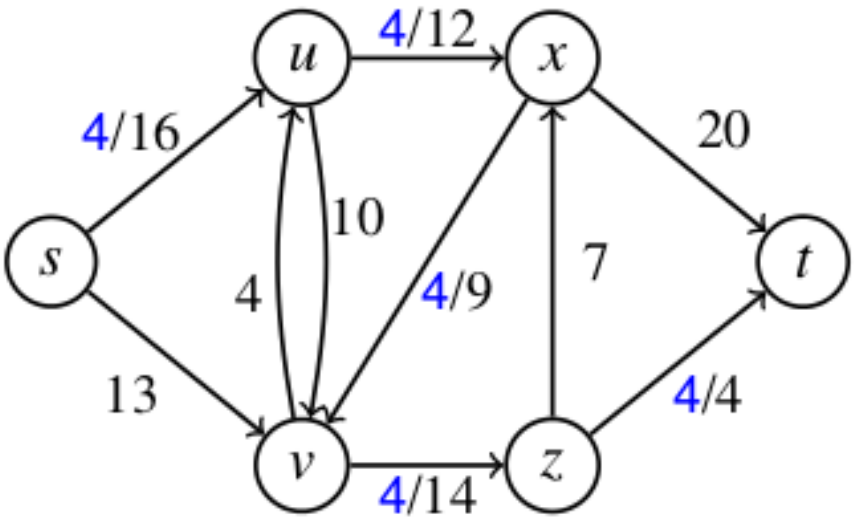


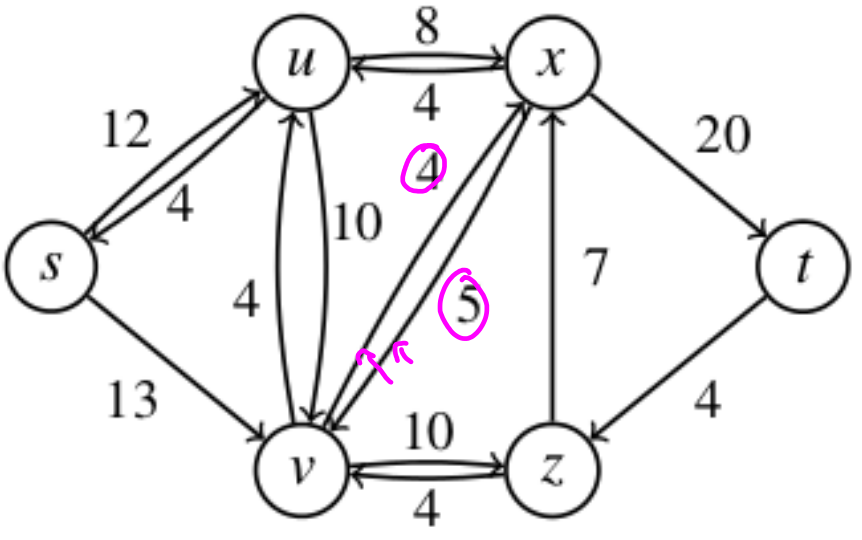
Algorithme de Ford-Fulkerson : exemple



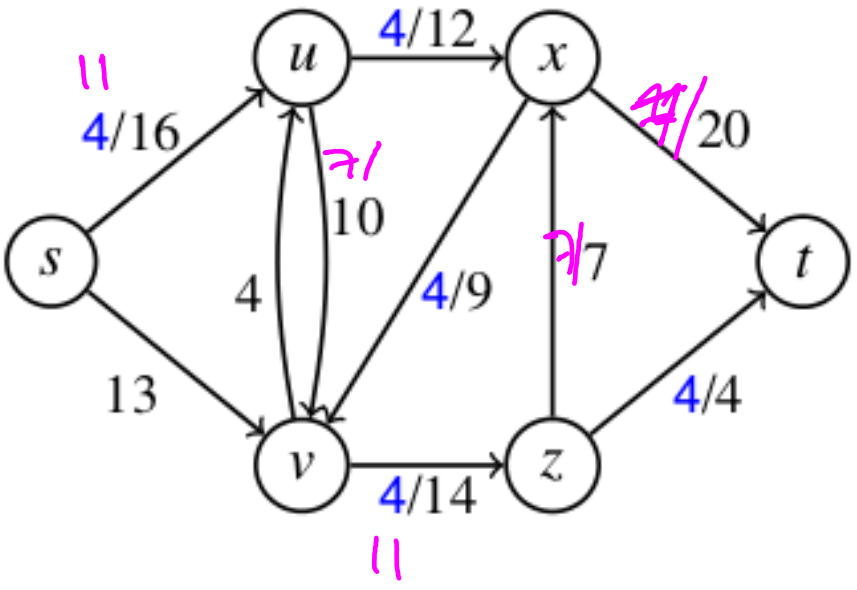


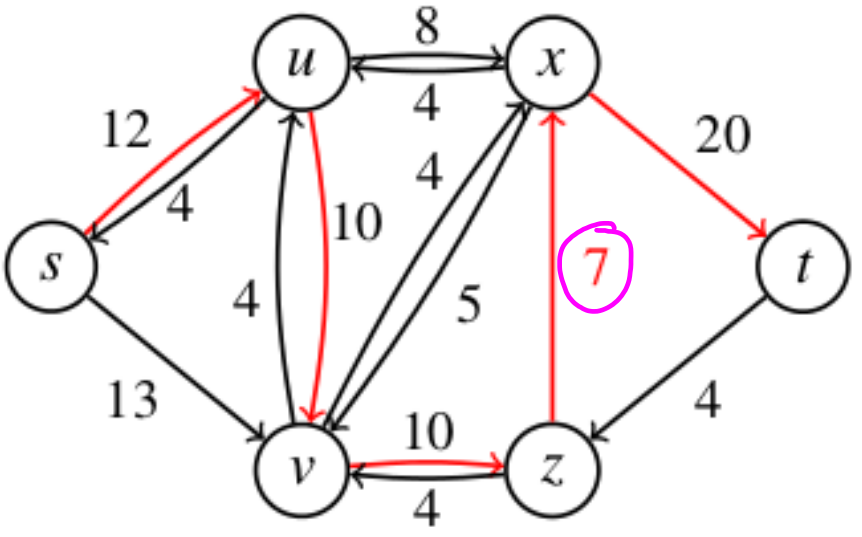
Algorithme de Ford-Fulkerson : exemple



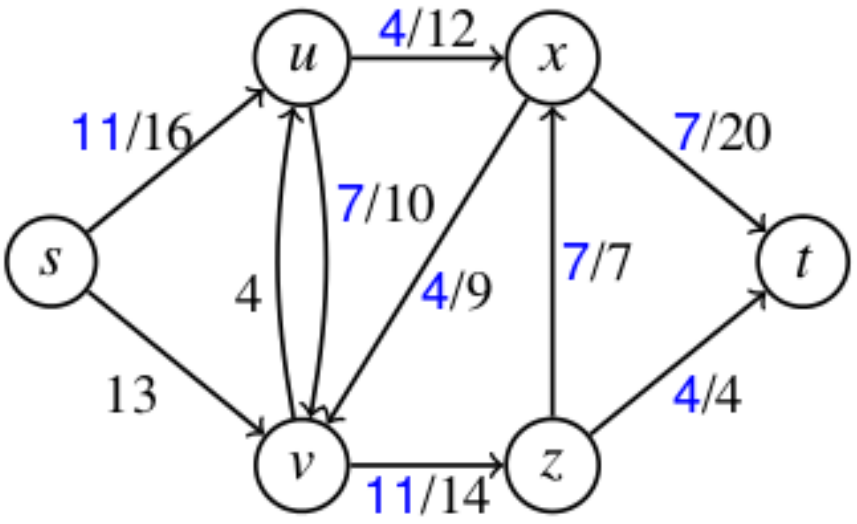


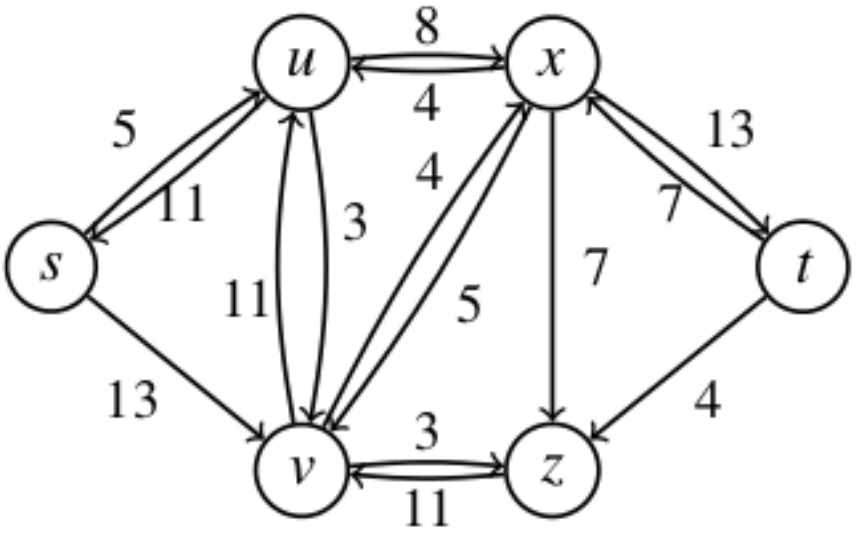
Algorithme de Ford-Fulkerson : exemple



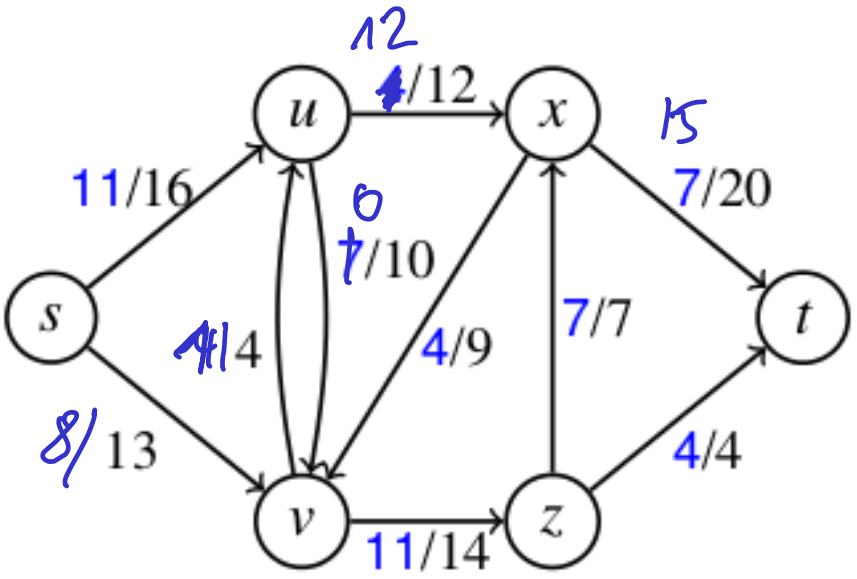


Algorithme de Ford-Fulkerson : exemple

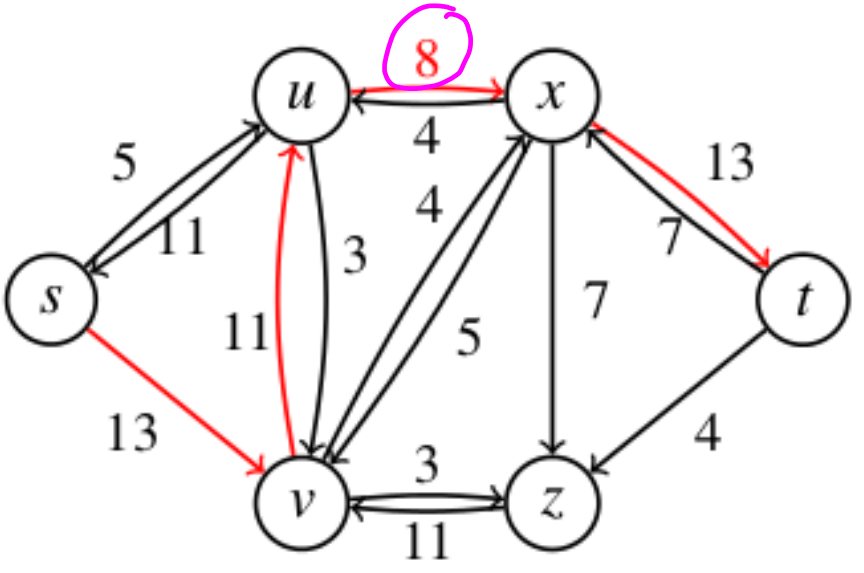




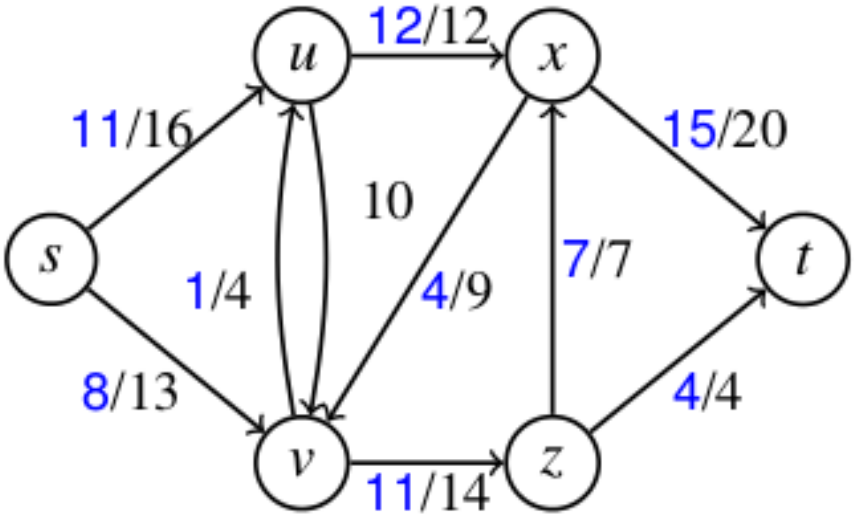
Algorithme de Ford-Fulkerson : exemple

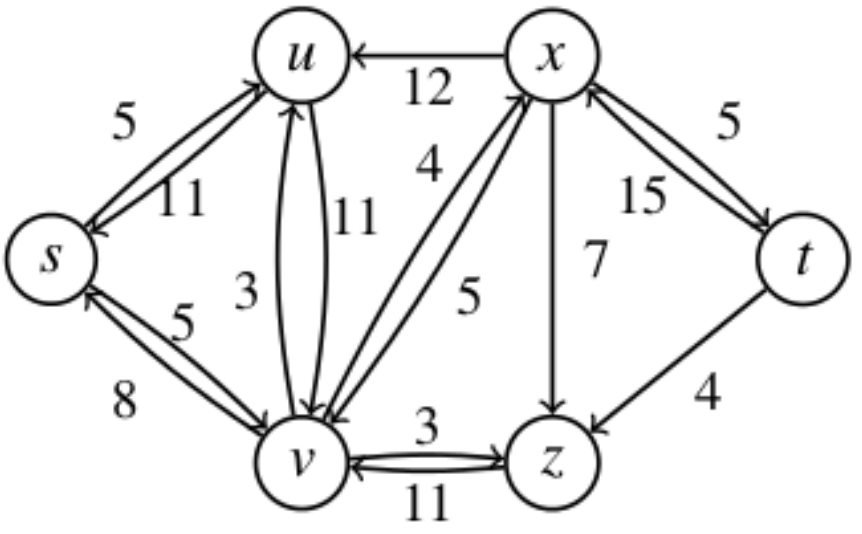


Cap. minimale

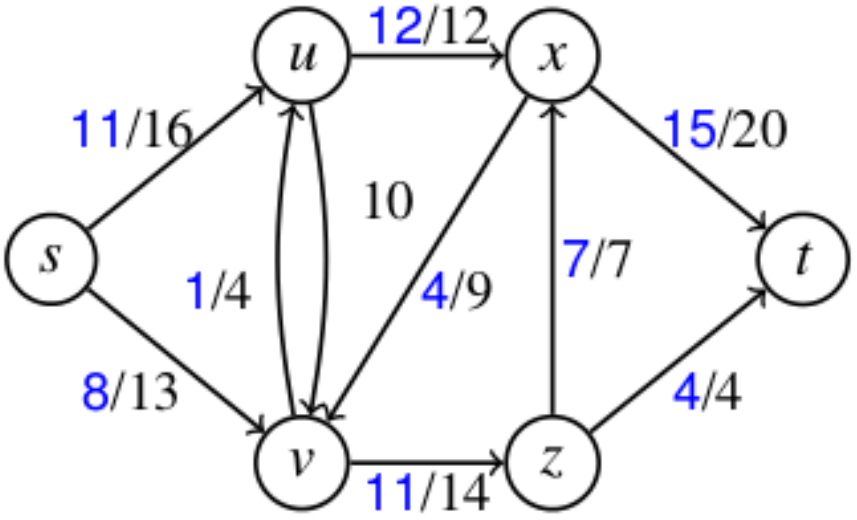


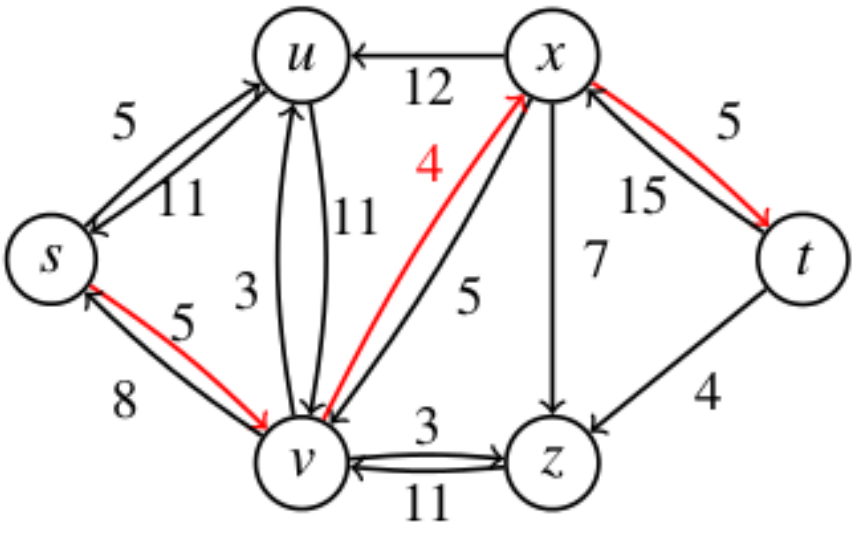
Algorithme de Ford-Fulkerson : exemple



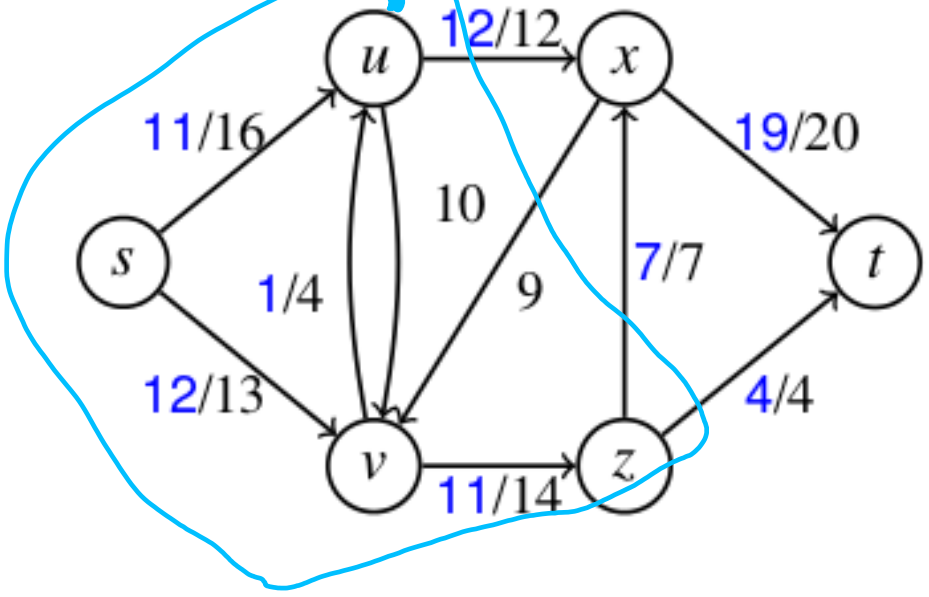


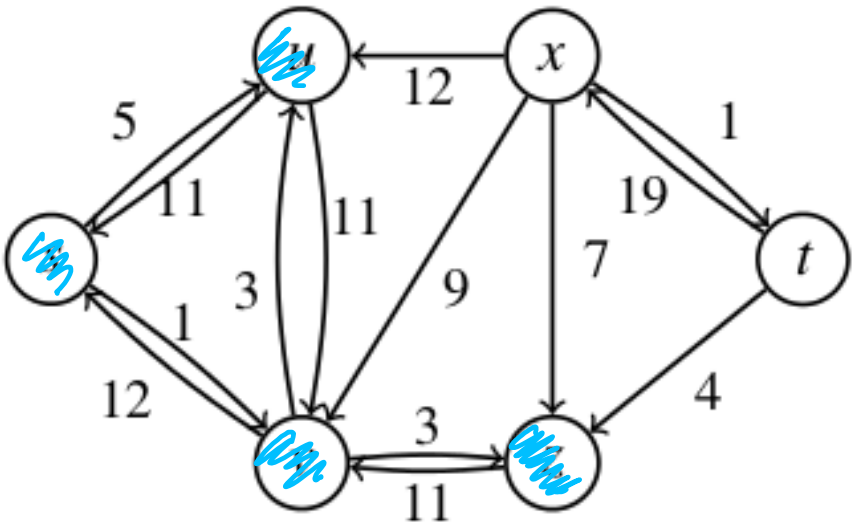
Algorithme de Ford-Fulkerson : exemple





Algorithme de Ford-Fulkerson : exemple

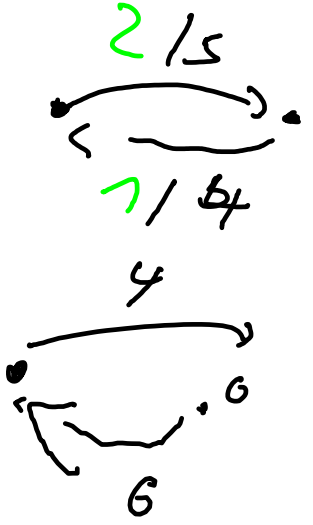




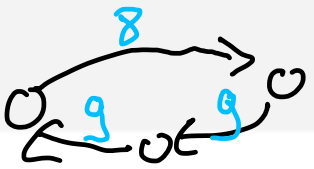
$cap(\delta^{out}(L)) = 23$

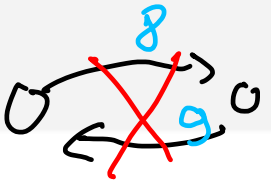
$f_{tot} = 23$

$L \subseteq V$



Dualité forte



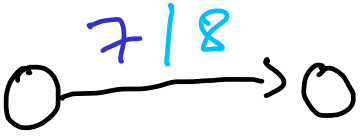


Theorem (Théorème du flot max et de la coupe min, dualité forte)

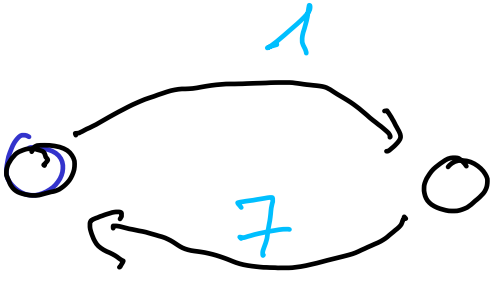
La valeur maximale d'un flot admissible de s à t est égale à la capacité d'une coupe minimale de $s - t$.

$f : A \to \mathbb{R}_{\geq 0}$

Graph résiduel $D(f)$



Réseau



Dualité forte

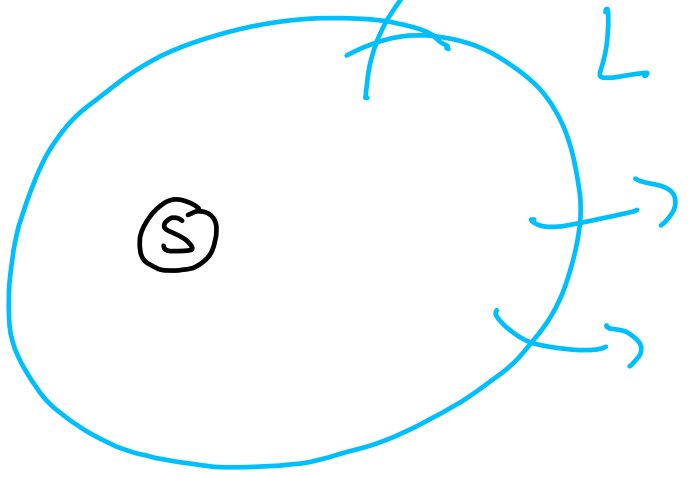
Theorem (Théorème du flot max et de la coupe min, dualité forte)

La valeur maximale d'un flot admissible de s à t est égale à la capacité d'une coupe minimale de $s - t$.

Démonstration : (Suppose que la val. max. est bornée

f : Flot maximal

=> max. flot existe.



$L \subseteq V$ sommets atteignable de s dans $D(f)$

$\Rightarrow t \notin L$ dès que f max.

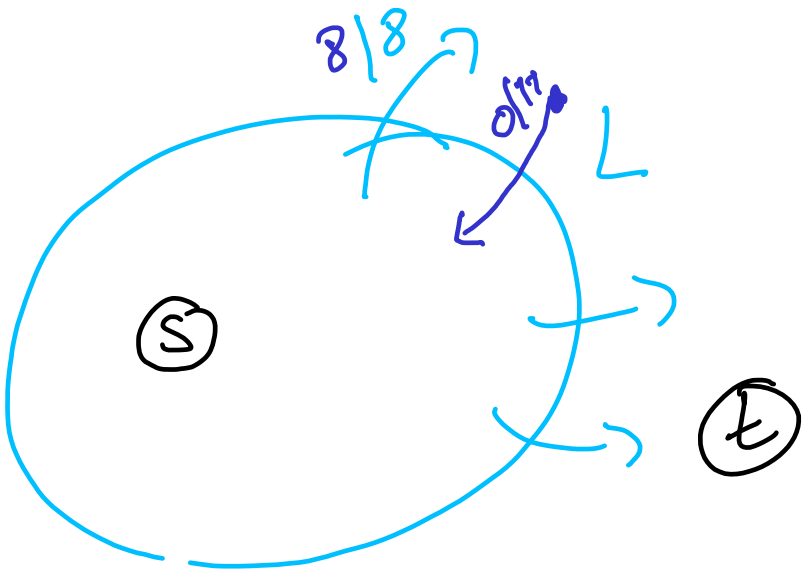
(contrairement on pourrait augmenter f encore !)

=> $\delta^{out}(L)$ une coupe s.t

Dualité forte

Theorem (Théorème du flot max et de la coupe min, dualité forte)

La valeur maximale d'un flot admissible de s à t est égale à la capacité d'une coupe minimale de $s - t$.



- excess(S) = valeur du flot f

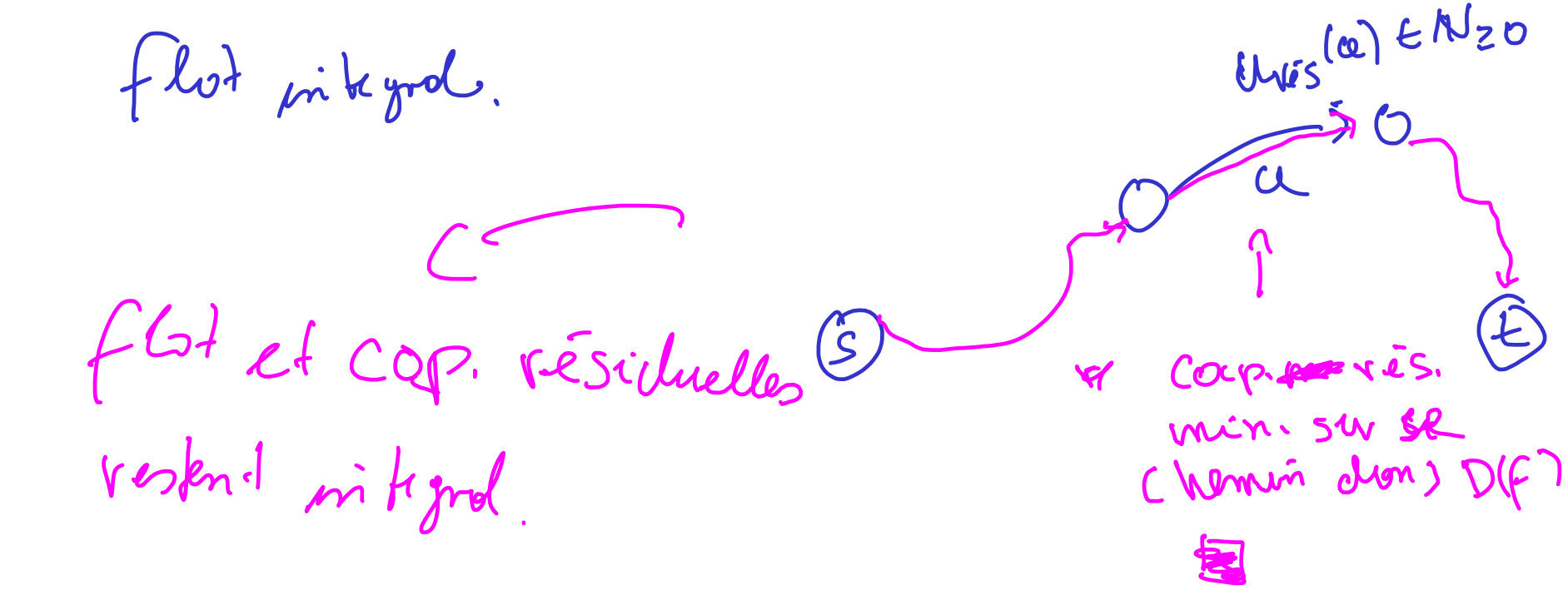
- excess(S) = - excess(L)
= valeur(f)
= $u(\delta^{out}(L))$

Dualité forte

Corollary (Théorème d'intégralité)

Si $u(a) \in \mathbb{N}$ pour tout $a \in A$, alors il existe un flot maximal entier ($f(a) \in \mathbb{N}$ pour tout $a \in A$).

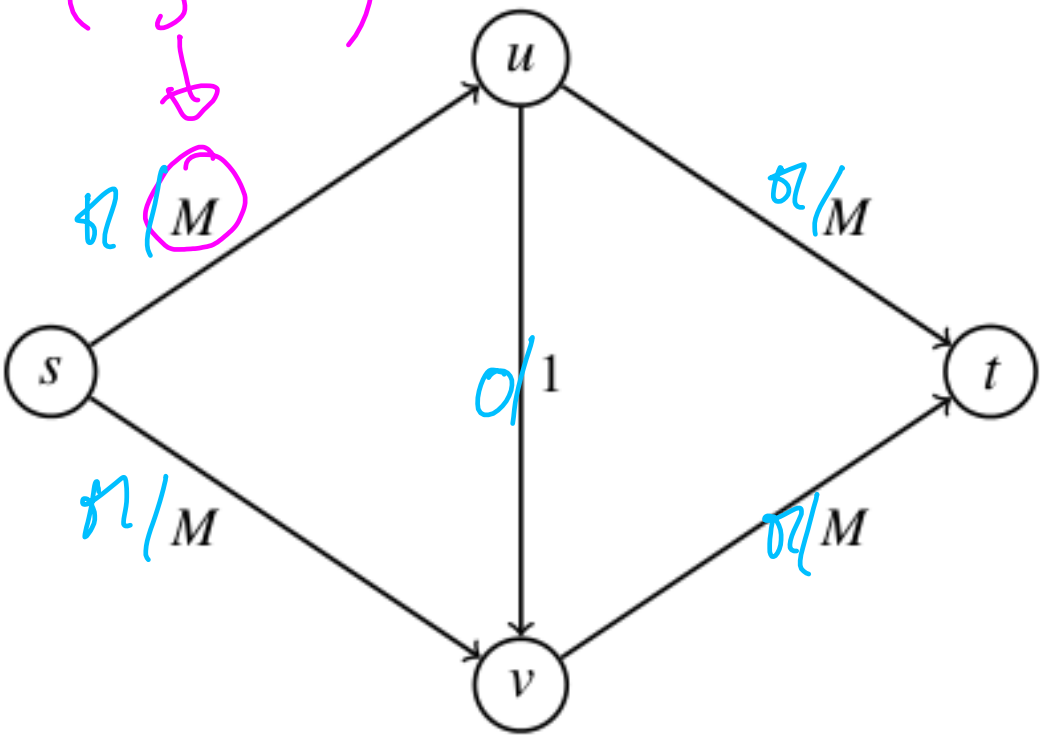
Dém. on augmente toujours avec un flot intégral.



L'algorithme de Ford-Fulkerson s'exécute-t-il en temps polynômial?

$\Theta(\log(n))$

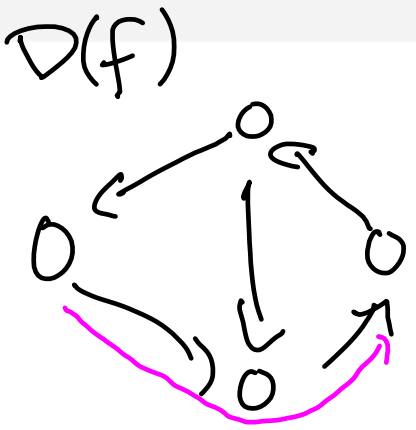
$n \in \mathbb{N}_{\geq 1}$

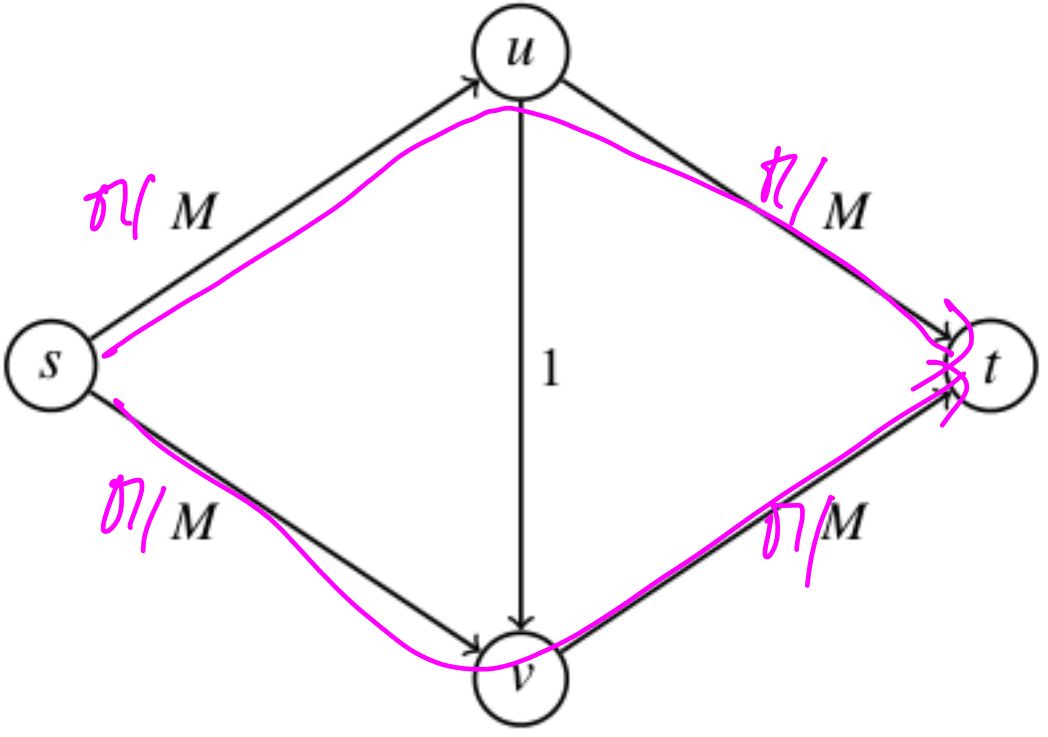


val

flot max $= 2n$.

L'algorithme de Ford-Fulkerson s'exécute-t-il en temps polynômial?

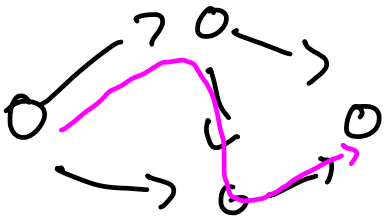




peut terminer en 2 étapes!

L'algorithme de Ford-Fulkerson s'exécute-t-il en temps polynômial?

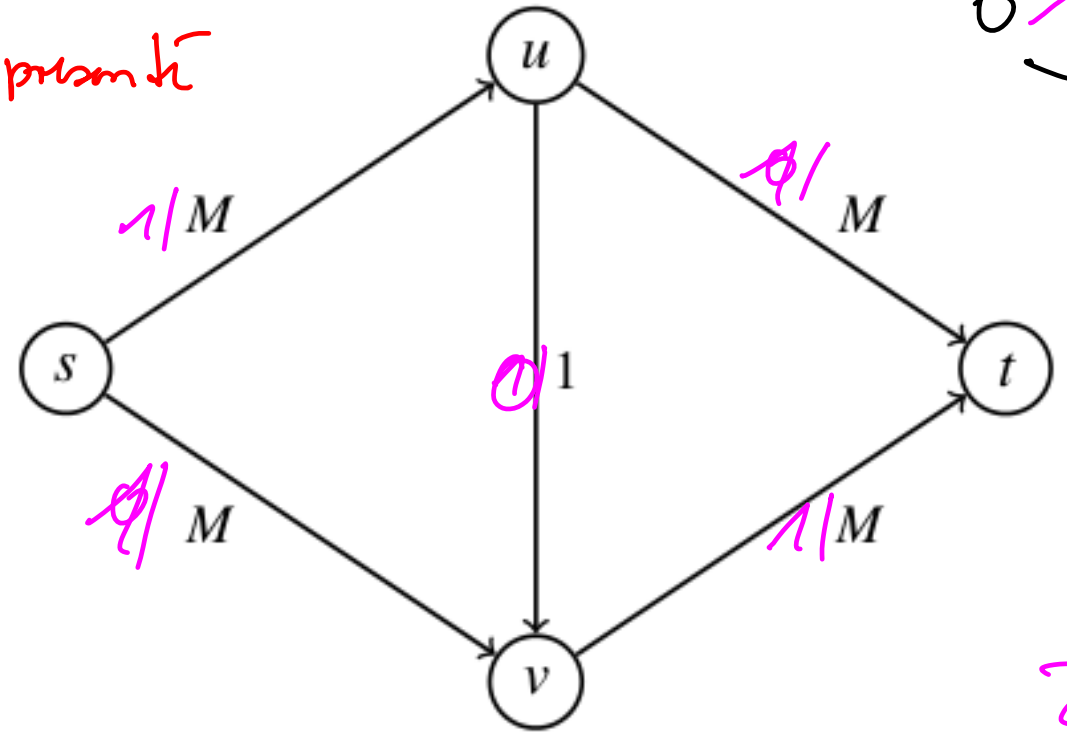
D(f)

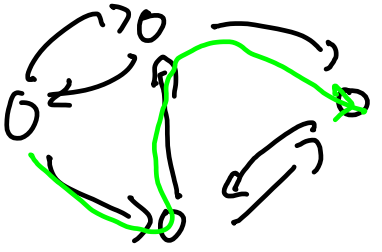


Réseau exponentiel

par $\Theta(\log(n))$

bits!!!





2

2^n itérations => exponentiel en $\log(n)$

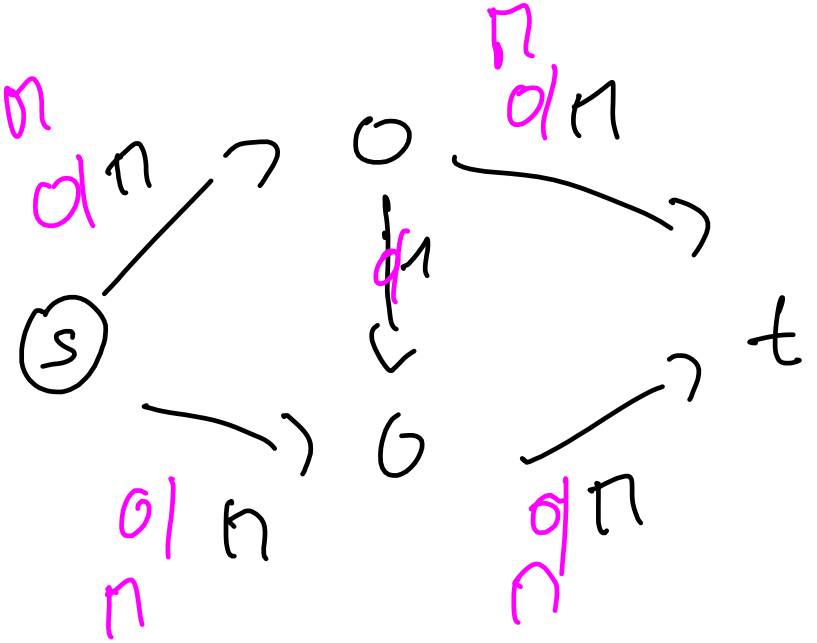
Lemme

Edmonds & Karp Idée : TROUVER CHEMIN PLUS court (non-pondéré) de s à t dans D(f)

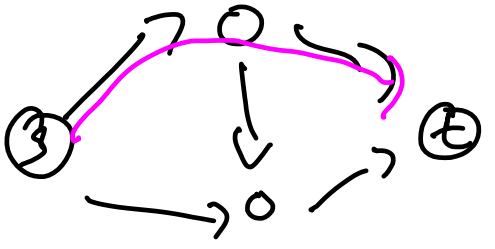
Soient $D = (V, A)$ un graphe orienté, $s, t \in V$ et $\mu(D)$ la longueur d'un plus court chemin de s à t. Soit $a(D)$ l'ensemble d'arcs contenu dans au moins un des plus courts chemins de s à t.

Lemme

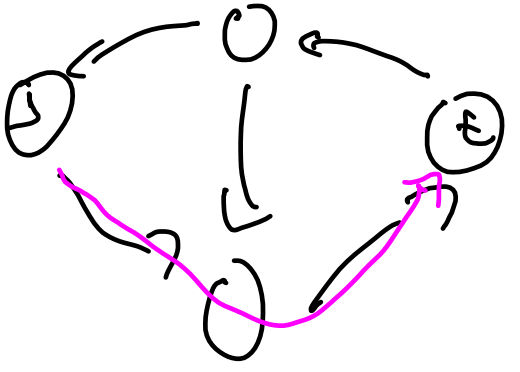
Soit $D = (V, A)$ un graphe orienté et $s, t \in V$. Définissons $D' = (V, A \cup a(D)^{-1})$. Alors $\mu(D) = \mu(D')$ et $a(D) = a(D')$.



D(f)



D(f')

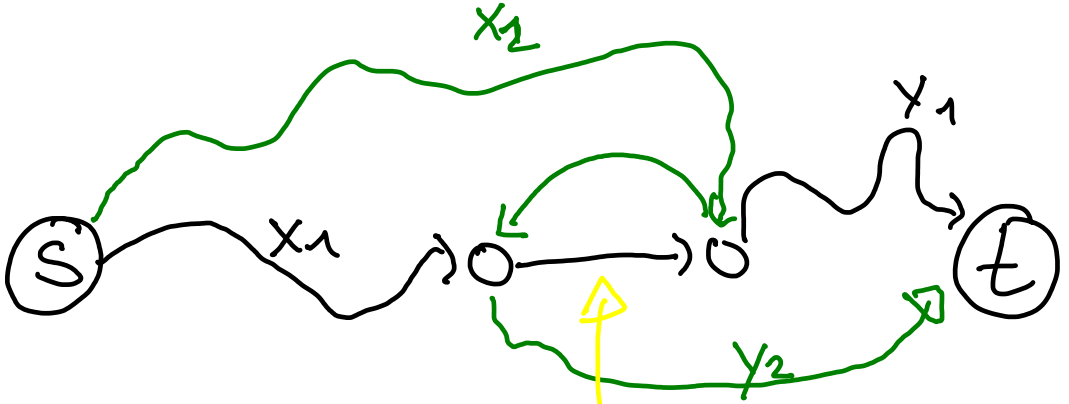


Lemme

Soient $D = (V, A)$ un graphe orienté, $s, t \in V$ et $\mu(D)$ la longueur d'un plus court chemin de s à t. Soit $a(D)$ l'ensemble d'arcs contenu dans au moins un des plus courts chemins de s à t.

Lemme

Soit $D = (V, A)$ un graphe orienté et $s, t \in V$. Définissons $D' = (V, A \cup a(D)^{-1})$. Alors $\mu(D) = \mu(D')$ et $a(D) = a(D')$.



longueur (P_1)

longueur (P_2)

Si $x_1 + y_1 \geq x_2 + y_2$

Alors $x_1 + y_2$ ou $x_2 + y_1$

$< x_1 + y_1 + 1$

Chemin plus court (non-pondéré) de s à t.

vue désespérante !!

Algorithme en temps polynômial

Théorème

Si à chaque itération nous choisissons un plus court chemin de s à t dans $D(f)$ comme chemin d'augmentation de flot, le nombre d'itérations est au plus $|V| \cdot |A|$.

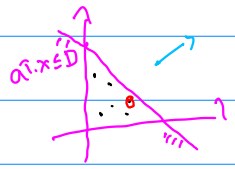
Programmation dynamique (Problème Sac à dos)

PLI : $\max c^T \cdot x$

$Ax \leq b$ Problème est

$x \in \mathbb{Z}^n$ NP-complète !

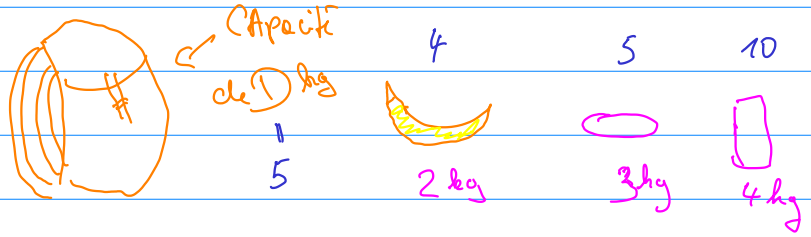
Le PLI le plus simple (?)



$$\max c^T x \qquad a \in \mathbb{N}^n$$

$$a^T x \leq D \qquad D \in \mathbb{N}$$

$$x \in \{0,1\}^n \qquad c \in \mathbb{N}_0^n$$



Solution optimale : Seulement Chocolat !

$a : (2,3,4)$, $D = 5$, $c = 4, 5, 10$

$$\max \; 4 \cdot x_1 + 5 \cdot x_2 + 10 \cdot y_3$$

$$2 \cdot x_1 + 3 \cdot x_2 + 4 \cdot x_3 \leq D$$

$$x_1, x_2, x_3 \in \{0,1\}$$

☹ Problème est aussi NP-hard.

Que faire ? But : Il y a en algo.

qui se déroule en temps polynomial dans l'encodage unaire.

$4 \rightarrow 100$ binaire

$\rightarrow 1111$ unaire

$100 \rightarrow \underbrace{1111\ldots1}_{100\times}$ unaire

Idée : Construire un graphe orienté sans cycles. tel que la solution optimal peut être récupérer d'un chemin plus court par dérivé.

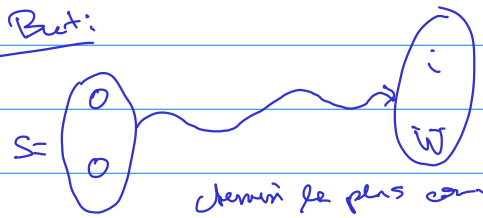
$0 \le i \le n$

Valeur $0 \le W \le \sum_{i=1}^{n} c_i$

polynomial en encoding unaire

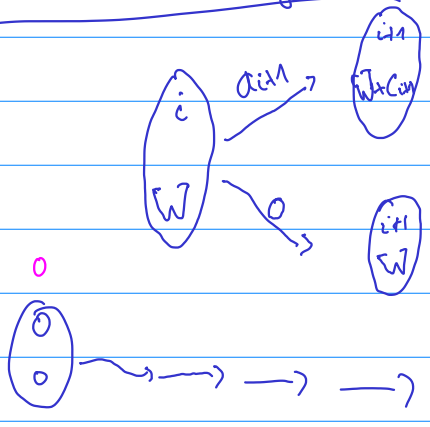
Nombre des sommets : $O(n \cdot \sum_{i=1}^{n} c_i)$ ← polynomial en encoding binaire

But :



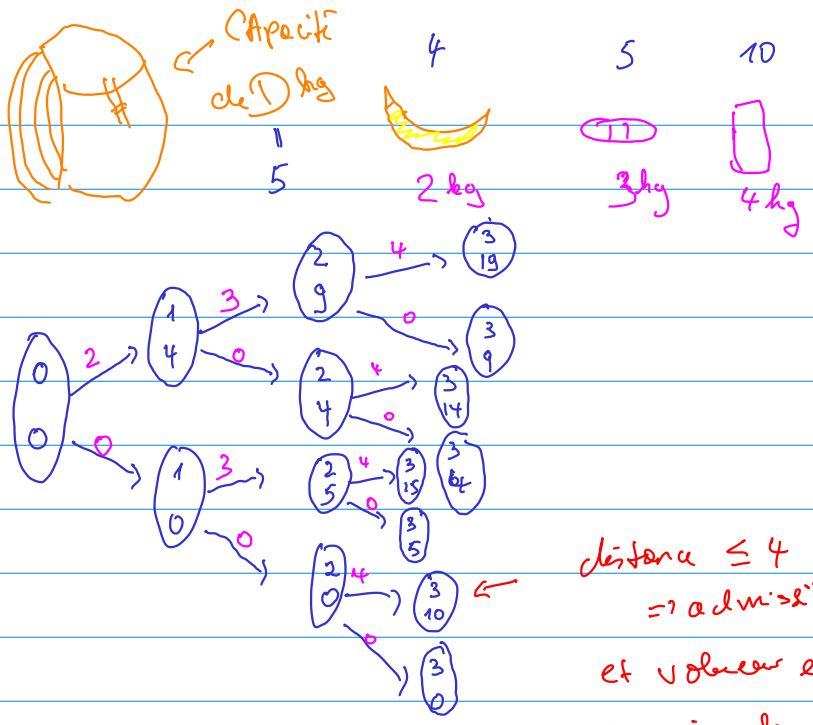
chemin le plus court possible devrait avoir une longueur ≃ poids min entre les premiers i objects qui donnent une volume de W

Construction de ce graphe !



Théorème : Le problème du Sac à dos peut être résolu en temps $O(n \cdot \sum_{i=1}^{n} c_i)$

Chemin encode un sous ensemble des premier i elements de valeur W.
La longueur de ce chemin est le poids de cet ensemble.


Capacité de 5 kg
5
4
2 kg
5
3 kg
10
4 kg
0 0
2 4
1 4
1 0
2 9
2 4
2 5
2 0
3 19
3 9
3 14
3 15
3 6
3 5
3 10
3 0
distance ≤ 4
⇒ admissible
et valeur est
maximale.