

Problèmes de flots

Vincent Mousseau

1. Réseau de transport et flots

1.1. Réseau de transport

- Définition : un réseau de transport de E à S est un graphe $G = (X, U)$ sans boucle, comporte une racine unique E (appelée entrée ou source) et une antiracine unique S (appelée sortie ou puit),

- à chaque arc $u = (x_i, x_j) \in U$ est associé :

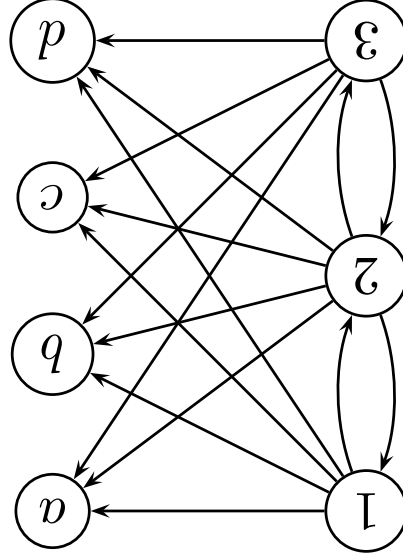
- une capacité notée $k(u)$ (ou $k(x_i, x_j)$) ou k_{ij}) représentant la quantité max. de matière pouvant circuler sur u , $k(u) \geq 0$,
- une borne notée $b(u)$ (ou $b(x_i, x_j)$ ou b_{ij}) représentant la quantité minimum de matière pouvant circuler sur l'arc u , $k(u) \geq b(u) \geq 0$,

- un coût unitaire noté $c(u)$ (ou $c(x_i, x_j)$ ou c_{ij}) représentant le coût d'acheminement d'une unité de matière sur l'arc u ,

Réseau de transport

Remarque 1 :

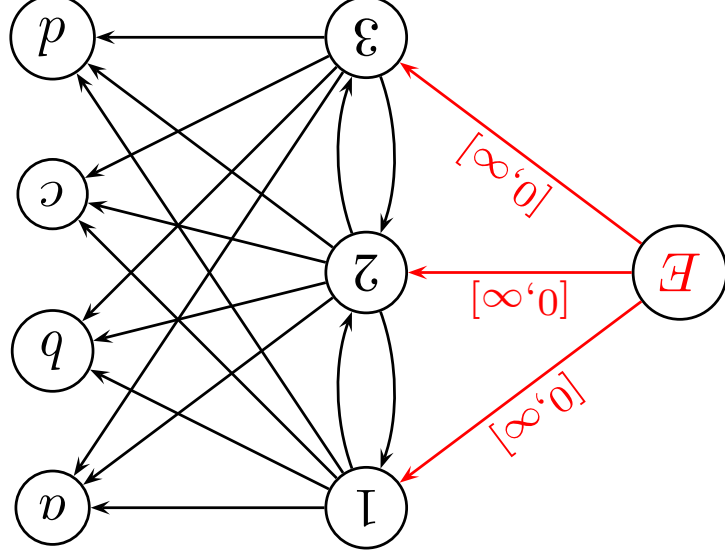
- si G comporte plusieurs racines e_1, e_2, \dots, e_n ,



Réseau de transport

Remarque 1 :

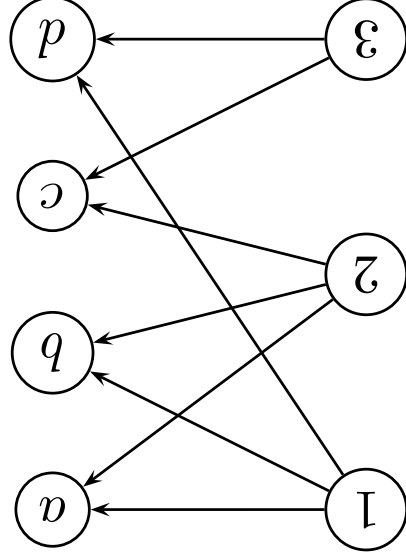
- si G comporte plusieurs racines e_1, e_2, \dots, e_n , on se ramène à un réseau en ajoutant à X un sommet E et à $U(E, e_i)$, $\forall i = 1, \dots, n$, avec $k(E, e_i) = +\infty$, $b(E, e_i) = 0$, $c(E, e_i) = 0$.



Réseau de transport

Remarque 2 :

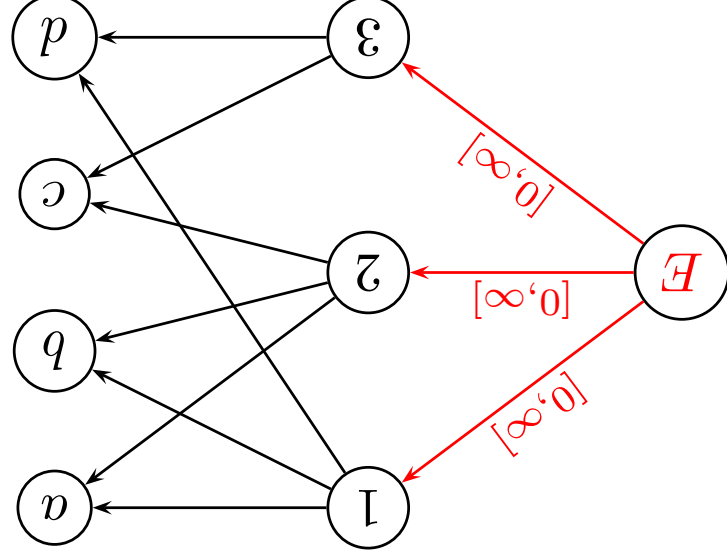
- si G n'a pas de racine,



Réseau de transport

Remarque 2 :

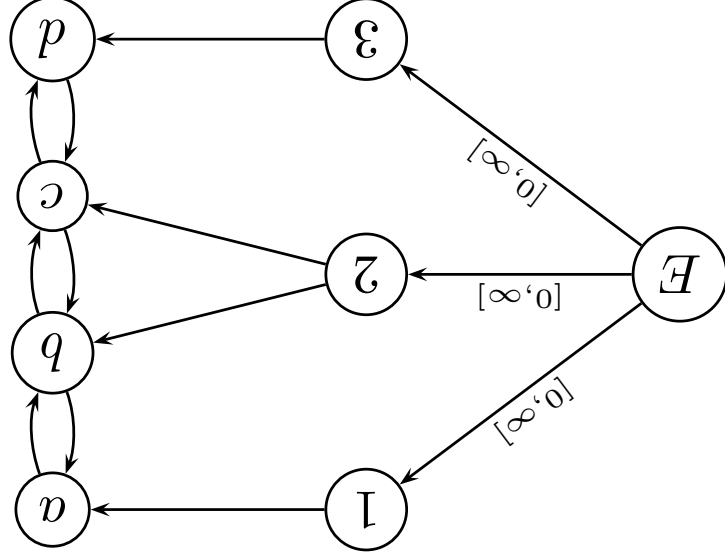
- si G n'a pas de racine, on ajoute un sommet E prédécesseur de tous les sommets x t.q. $\Gamma^{-1}(x) = \emptyset$, $k(E, x) = +\infty$, $b(E, x) = 0$, $c(E, x) = 0$.



Réseau de transport

Remarque 3 :

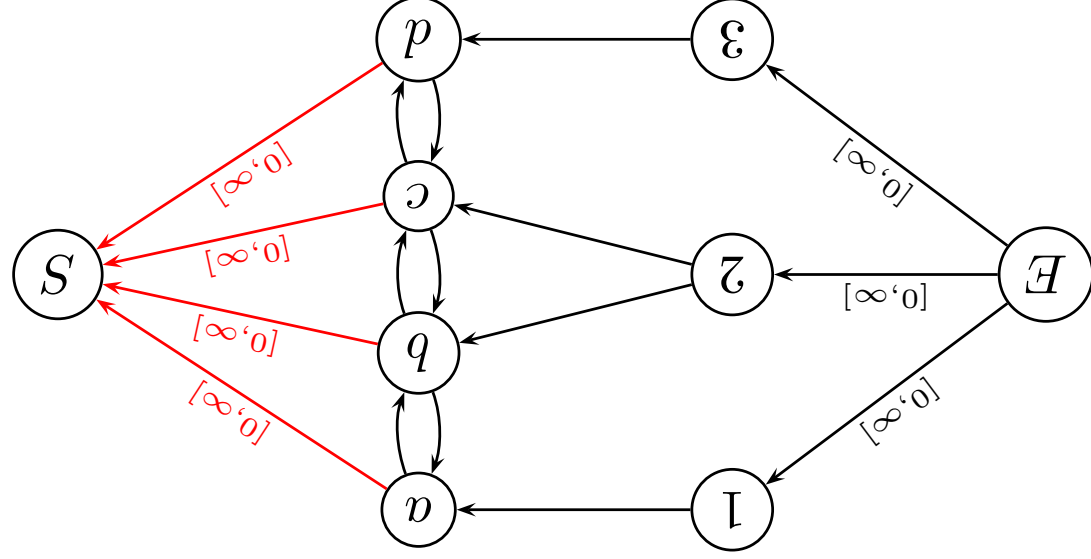
- si G comporte plusieurs antiracines s_1, s_2, \dots, s_n ,



Réseau de transport

Remarque 3 :

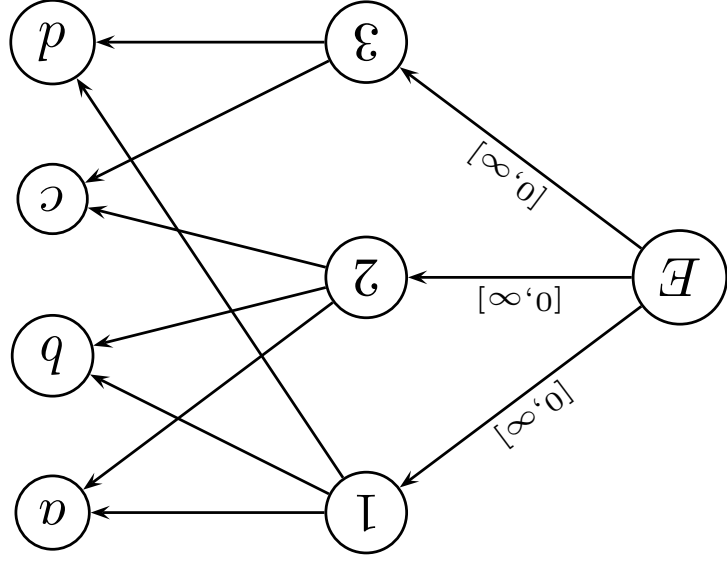
- si G comporte plusieurs antiracines s_1, s_2, \dots, s_n , on se ramène à un réseau en ajoutant à X un sommet S et à $U(s_i, S)$, $\forall i = 1, \dots, n$, avec $k(s_i, S) = +\infty$, $b(s_i, S) = 0$, $c(s_i, S) = 0$.



Réseau de transport

Remarque 4 :

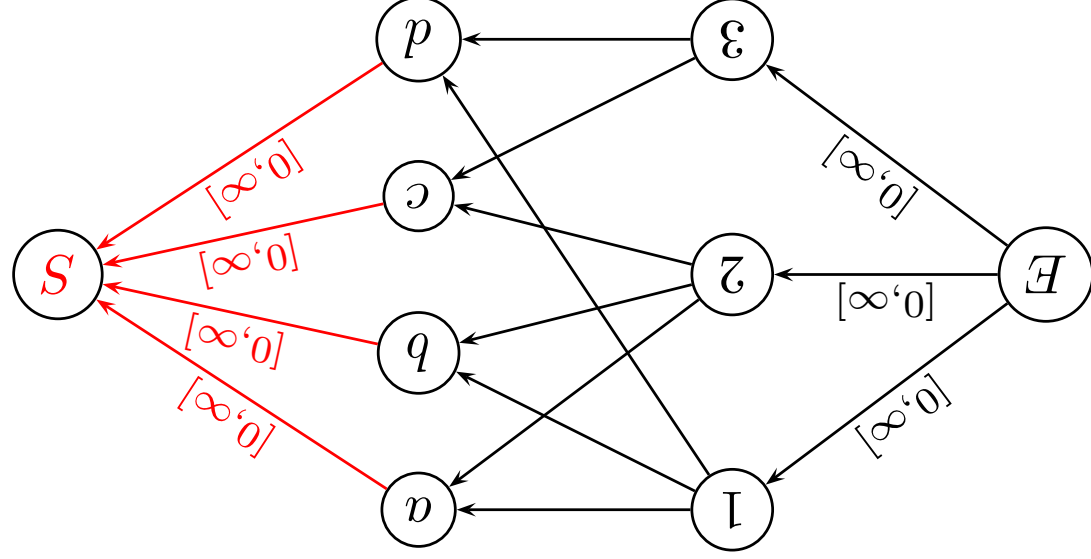
- si G n'a pas d'antiracine,



Réseau de transport

Remarque 4 :

- si G n'a pas d'antiracine, on ajoute un sommet S successeur de tous les sommets x t.q. $\Gamma(x) = \emptyset$, $k(x, S) = +\infty$, $b(x, S) = 0$, $c(x, S) = 0$.



Exemple

Transport d'un produit d'entrepôts à des clients

- 3 dépôts (1, 2 et 3), quantité en stock : 45, 25 et 25 resp.,
- 4 clients (a , b , c et d), demande : 30, 10, 20 et 30 resp.,
- Les limitations en matière de transport de produit d'un entrepôt à un client sont définies par :

	a	b	c	d
1	10	15	-	20
2	20	5	5	-
3	-	-	10	10

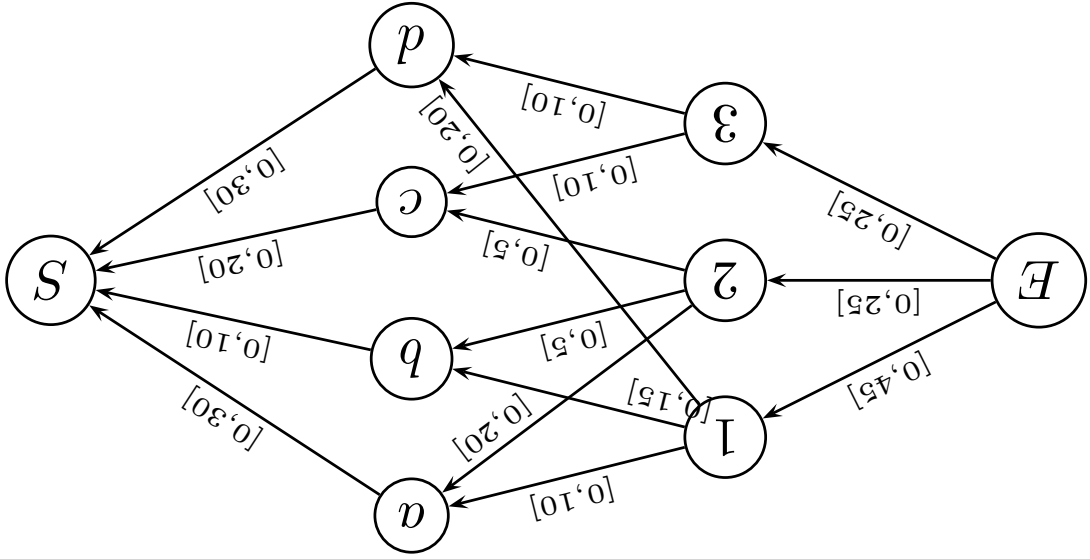
- Questions :

Peut-on satisfaire toutes les demandes ? Si oui, comment ?
 Sinon comment les satisfaire au mieux ?

Modélisation

Remarques :

- $X = \{1, 2, 3\} \cup \{a, b, c, d\} \cup \{E\} \cup \{S\}$,
- U représente :
 - les possibilités de transport d'un d'entrepôt à un client,
 - les arcs ajoutés pour constituer la racine E et l'antiracine S .
- le flux représente :
 - le déstockage de produits à un dépôt,
 - le transport de produits d'un dépôt à un client,
 - l'arrivée de produit chez un client.



existence d'une	existence d'une	existence d'une
disponibilité au	liaison dépôt i	demande chez le
dépôt i	client j	client j
quantité	quantité	quantité
de	de	de
matière	matière	matière
provenant	du	provenant
dépôt i	liaison dépôt i	dépôt i
x	n	x

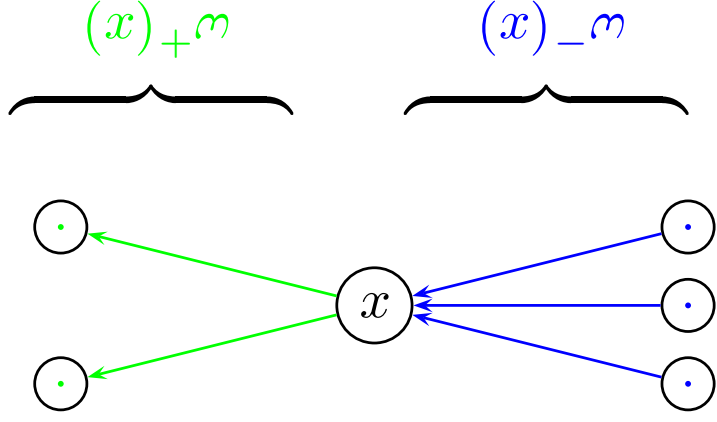
On note sur chaque arc u l'intervalle $[b(u), k(u)]$ ainsi que $\phi(u)$.

1.2. Flot, flux

- La quantité qui transite sur chaque arc $u \in U$ s'appelle le **flux** associé à (ou passant par) l'arc u . On note $\phi(u)$,

- Definition : un **flot** Φ défini sur un réseau de transport est un vecteur $\Phi = (\phi_1, \phi_2, \dots, \phi_m)$ avec $\phi_i = \phi(u_i)$, $i = 1, 2, \dots, m$ vérifiant la loi de conservation (loi de Kirshhoff), *i.e.*,

$$\sum_{u \in \omega^-(x)} \phi(u) = \sum_{u \in \omega^+(x)} \phi(u), \quad \forall x \in X \setminus \{E, S\}$$



1.2. Flot, flux

- Dans certains cas on ajoutera un arc "retour" (S, E) de sorte que la loi de conservation soit vérifiée $\forall x \in X$,
- Un **flot réalisable** (ou compatible) est un flot vérifiant $b(u) \leq \phi(u) \leq k(u), \forall u \in U$.

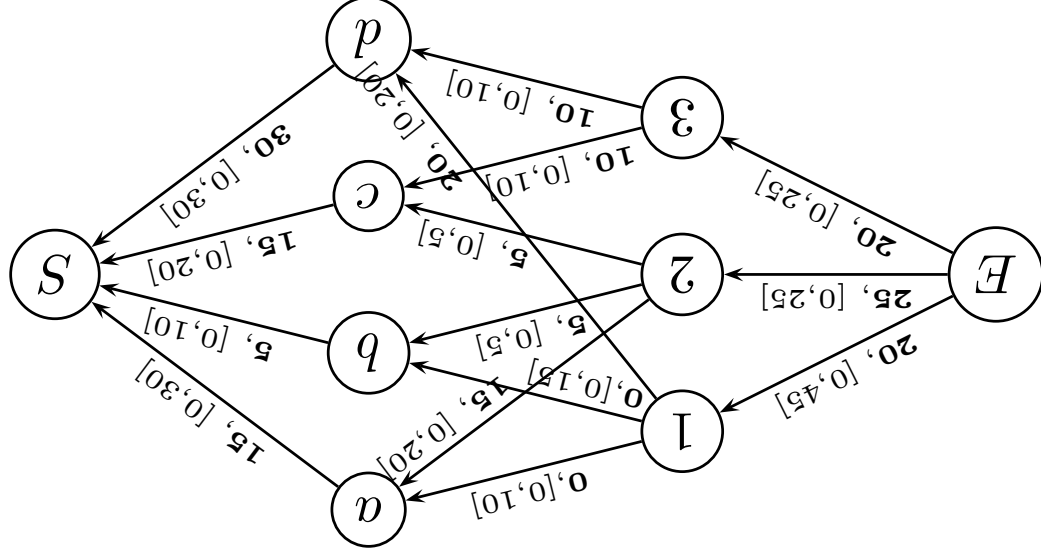
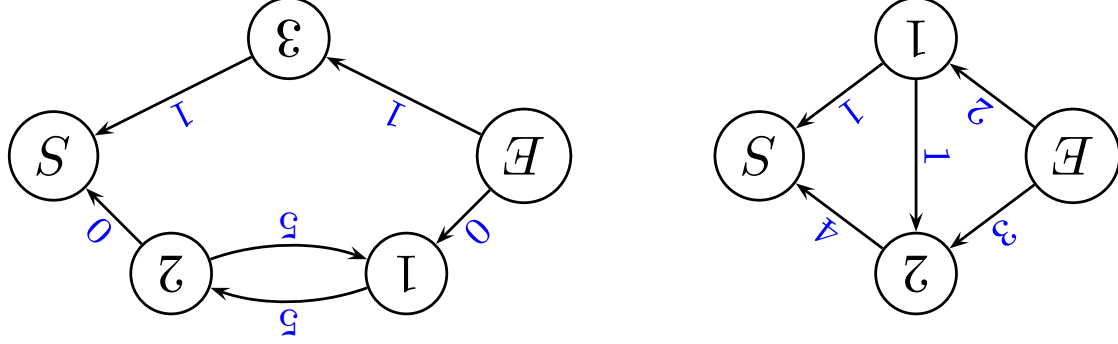


Figure 1 : exemple de flot réalisable

1.2. Flot, flux

- $v(\Phi)$, la **valeur d'un flot** Φ représente la quantité totale de matière circulant sur le réseau. Elle est définie par :

$$v(\Phi) = \sum_{n \in \omega^+(E)} \phi(n) = \sum_{n \in \omega^-(S)} \phi(n)$$



- Le coût d'un flot représente le coût associé au transport de l'ensemble de la matière sur le réseau, *i.e.*, $c(\Phi) = \sum_{n \in U} c(n) \cdot \phi(n)$,
- si le coût sur chaque arc des exemples est égal à 1,

$$c(\Phi_1) = (2 \times 1) + (3 \times 1) + (1 \times 1) + (1 \times 1) + (4 \times 1) = 11$$

$$c(\Phi_2) = (1 \times 1) + (1 \times 1) + (5 \times 1) + (5 \times 1) + (1 \times 1) = 12$$

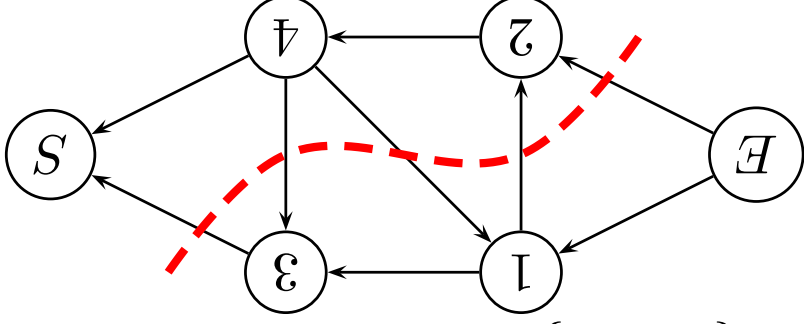
1.3. Coupe

- Définition : Soit $A \subset X$, t.q. $E \in A, S \notin A$. On appelle **coupe engendrée par A** (notée $\omega(A)$) l'ensemble des arcs dont une des extrémités appartient à A et l'autre à $X \setminus A$.

- On distingue :

- $\omega_+(A) = \{n \in \omega(A) \text{ t.q. l'extrémité init. de } n \text{ appartient à } A\}$,
- $\omega_-(A) = \{n \in \omega(A) \text{ t.q. l'extrémité term. de } n \text{ appartient à } A\}$

- Exemple : $A = \{E, 1, 3\} \subset X$.



- La coupe engendrée par A est :

$$\omega(A) = \omega_+(A) \cup \omega_-(A) = \{(e, 2), (1, 2), (3, s)\} \cup \{(4, 1), (4, 3)\}$$

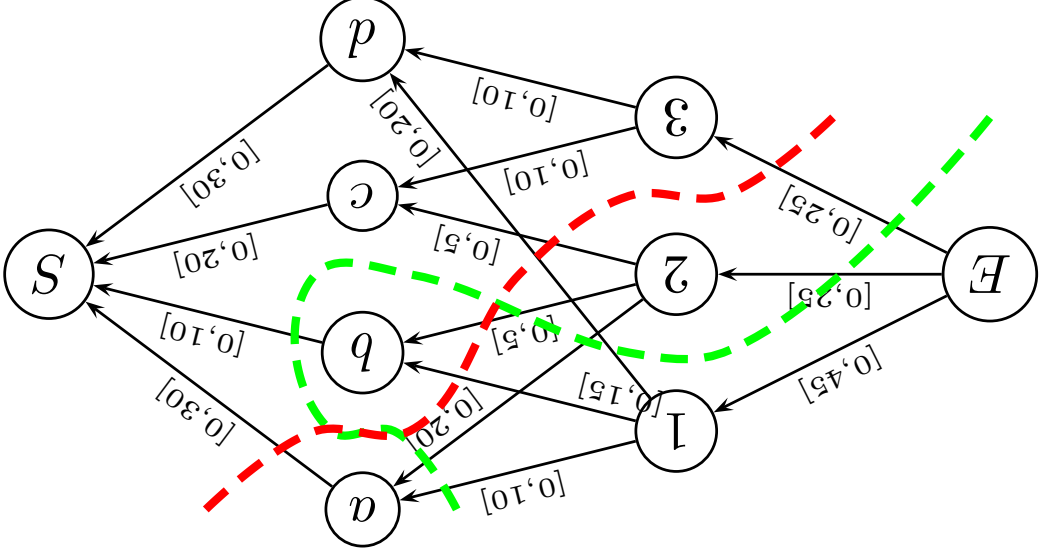
1.3. Coupe

- Définition : On appelle **valeur d'une coupe** $v(\omega(A))$ (notée $v(\omega(A))$) la quantité de matière pouvant circuler sur les arcs de la coupe, *i.e.*,

$$v(\omega(A)) = \sum_{n \in \omega^+(a)} k(n) - \sum_{n \in \omega^-(a)} b(n)$$

1.3. Coupe

- Example : $A = \{E, 1, 2, a\}$, $B = \{E, 1, b\}$.



$$\begin{aligned} \omega(A) &= \omega_+(A) \cup \omega_-(A) = \{(E, 3), (1, b), (1, d), (2, b), (2, c), (a, S)\} \cup \emptyset = 115 \\ \omega(B) &= \omega_+(B) \cup \omega_-(B) = \{(E, 2), (E, 3), (1, a), (1, d), (b, S)\} \cup \emptyset = 90 \\ v(\omega(A)) &= 115 \\ v(\omega(B)) &= 90 \end{aligned}$$

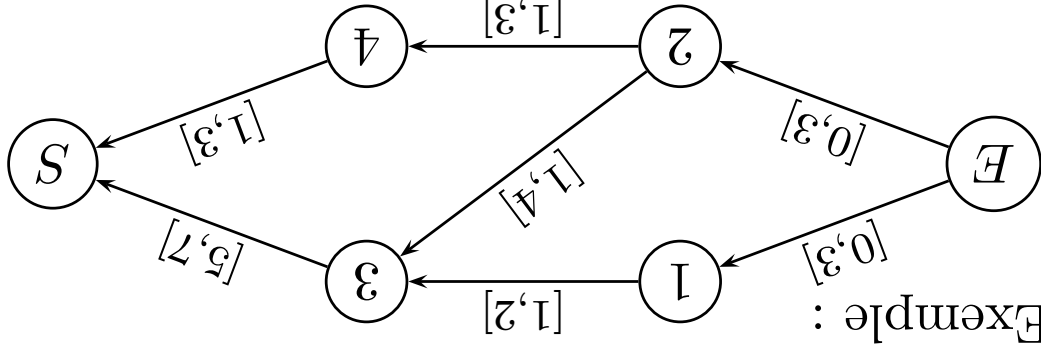
1.3. Coupe

- Chaque coupe constitue un “goulot d'étranglement” qui limite la valeur du flot.
- Théorème (relation entre flot et coupe): soit Φ un flot réalisable quelconque, soit un ensemble de sommets $A \subset X$ tel que $E \in A, S \notin A$, on a : $v(\Phi) \leq v(\omega(A))$
Interprétation : la valeur de tout flot est majorée par la valeur d'une coupe quelconque ; ceci est vrai en particulier pour le flot maximum,
- La notion de coupe correspond bien à l'idée de goulet d'étranglement.

2. Différents types de problèmes de flot

- 2.1. Détermination d'un flot réalisable sur un réseau de transport
- un **flot réalisable** est un flot Φ vérifiant $b(u) \leq \phi(u) \leq k(u)$, $\forall u \in U$ et $\sum_{u \in \omega^-(x)} \phi(u) = \sum_{u \in \omega^+(x)} \phi(u)$, $\forall x \in X \setminus \{E, S\}$,
 - un réseau de transport n'admet pas nécessairement de flot réalisable,

• Exemple :



on a $\phi(2, 4) \geq 1$ et $\phi(E, 2) \leq 3 \Rightarrow \phi(2, 3) \leq 2$
 or $\phi(1, 3) \leq 2$ donc $\phi(3, S) \leq 4 \leq b(3, S)$

Flot réalisable sur un réseau de transport

- *Cas particulier important :*
 si le réseau de transport ne comporte aucune borne
 $(b(u) = 0, \forall u \in U)$ alors il existe toujours un flot réalisable, i.e.,
 le flot nul Φ_0 t.q. $\phi(u) = 0, \forall u \in U$.

2.2. Détermination d'un flot maximum

- Objectif : faire transiter dans le réseau de transport la plus grande quantité de matière,

- Rechercher parmi tous les flots réalisables, le flot Φ^* tel que $v(\Phi^*)$ soit maximale,

- Ce problème peut s'écrire sous la forme d'un programme linéaire:

$$\begin{aligned} \max \quad & \sum_{n \in \omega^+(E)} \phi(n) \\ \text{s.c.} \quad & \sum_{n \in \omega^-(x)} \phi(n) = \sum_{n \in \omega^+(x)} \phi(n), \quad \forall x \neq E, S \\ & b(n) \leq \phi(n) \leq k(n), \quad \forall n \in U \\ & \phi(n) \geq 0 \end{aligned}$$

2.3. Détermination d'un flot de coût minimum

- Objectif : rechercher parmi les flots d'une valeur v donnée, celui (ou ceux) qui minimise(nt) le coût,
- Ce problème peut s'écrire sous la forme d'un programme linéaire:

$$\begin{aligned} & \min \sum_{n \in U} c(n) \cdot \phi(n) \\ \text{s.c.} \quad & \sum_{n \in \omega^-(x)} \phi(n) = \sum_{n \in \omega^+(x)} \phi(n), \quad \forall x \neq E, S \\ & b(n) \leq \phi(n) \leq k(n), \quad \forall n \in U \\ & \sum_{n \in \omega^+(E)} \phi(n) = v \\ & \phi(n) \geq 0 \end{aligned}$$

- *Cas particulier* : détermination d'un flot maximum à coût minimum ($v = v(\Phi^*)$)

3. Recherche d'un flot de valeur maximale

3.1. Principe général de l'algorithme de Ford-Fulkerson

- Principe général : on part d'un flot réalisable (flot nul si $b(u) = 0, \forall u \in U$) que l'on améliore itérativement,

- Définitions :

- un arc $u \in U$ est dit **saturé** si $\phi(u) = k(u)$,
- un flot Φ est dit **complet** si tout chemin μ de E vers S comporte au moins un arc saturé,

- *Remarque* : un flot non complet ne peut pas être maximum

- Φ maximum $\Leftrightarrow \Phi$ complet,

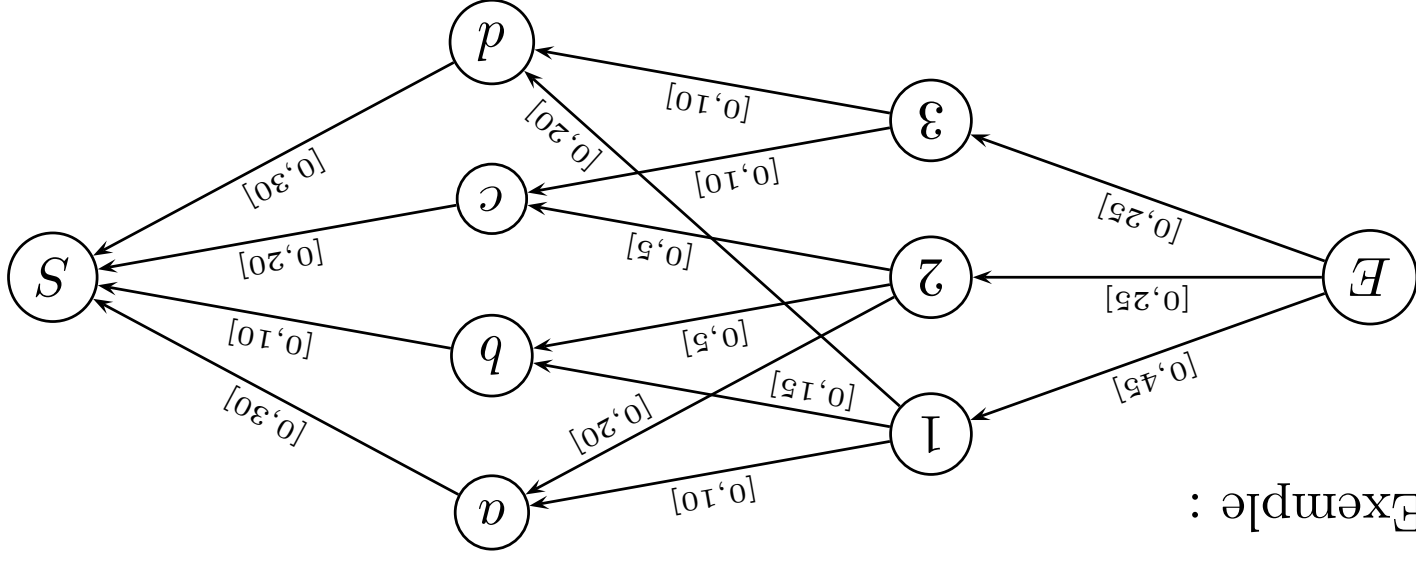
- mais Φ complet $\not\Leftrightarrow \Phi$ maximum ,

- Construire un flot complet lors de la première étape de

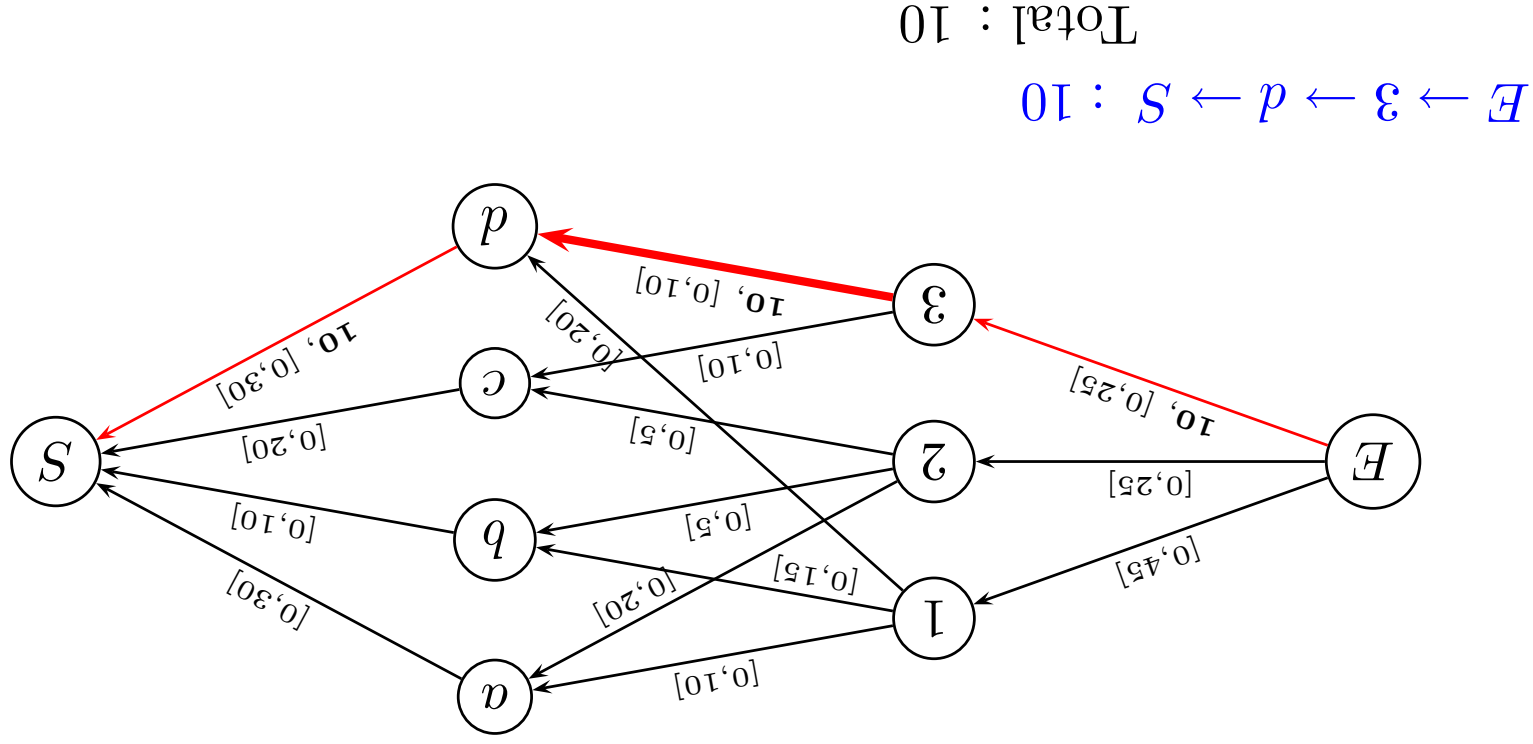
l'algorithme \rightarrow on part d'un flot complet au lieu de Φ_0 ,

3.2. Construction d'un flot complet

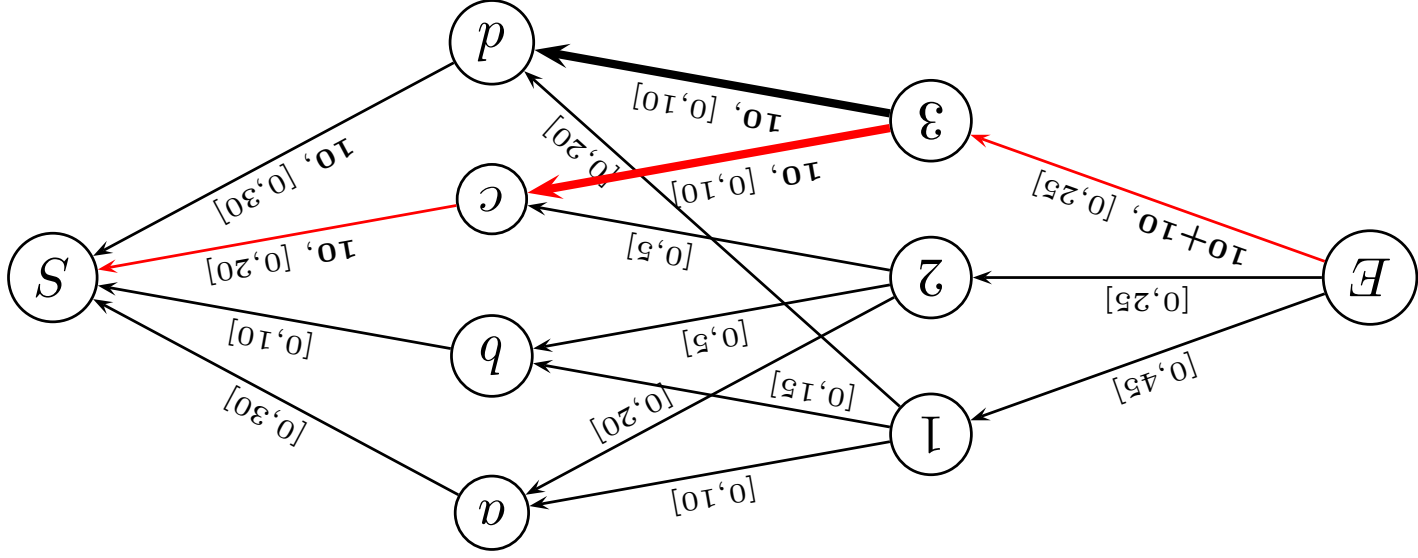
- Partir d'un flot réalisable (flot nul si $b(u) = 0, \forall u \in U$),
- Examiner tous les chemins de E à S de façon systématique,
- Pour chaque chemin μ de E à S , faire passer un flot égal à la capacité résiduelle minimale des arcs de μ .
- Exemple :



Construction d'un flot complet



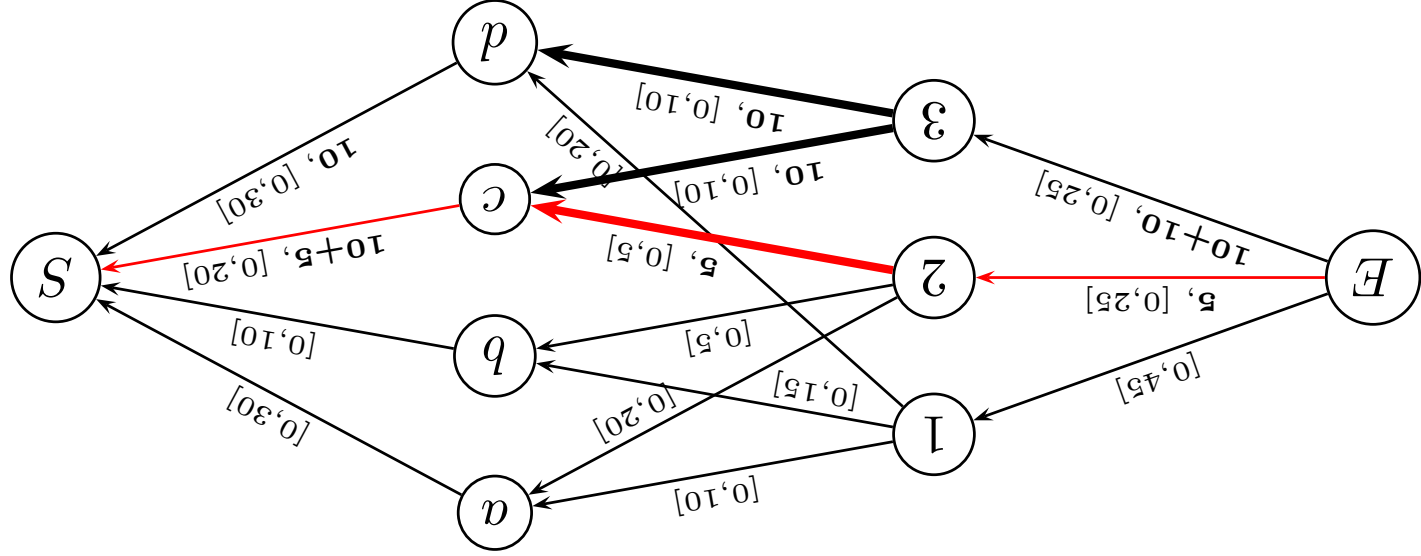
Construction d'un flot complet



$E \rightarrow 3 \rightarrow d \rightarrow S : 10$
 $E \rightarrow 3 \rightarrow c \rightarrow S : 10$

Total : 20

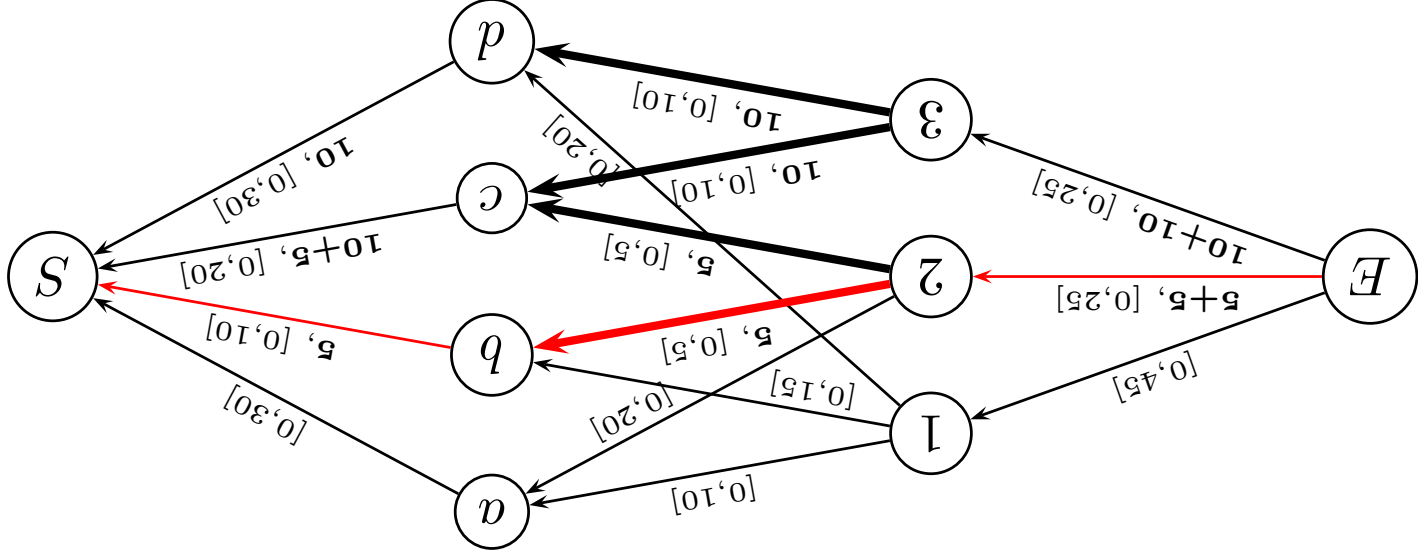
Construction d'un flot complet



Total : 25

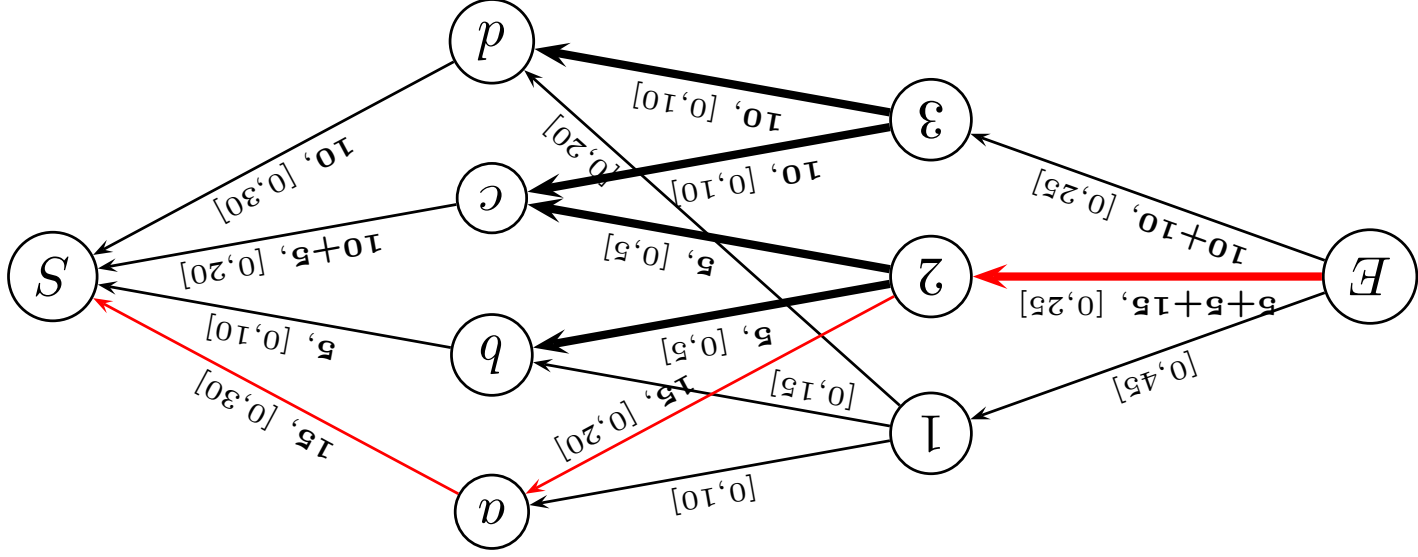
$E \rightarrow 3 \rightarrow d \rightarrow S : 10,$ $E \rightarrow 3 \rightarrow c \rightarrow S : 10$
 $E \rightarrow 2 \rightarrow c \rightarrow S : 5$

Construction d'un flot complet



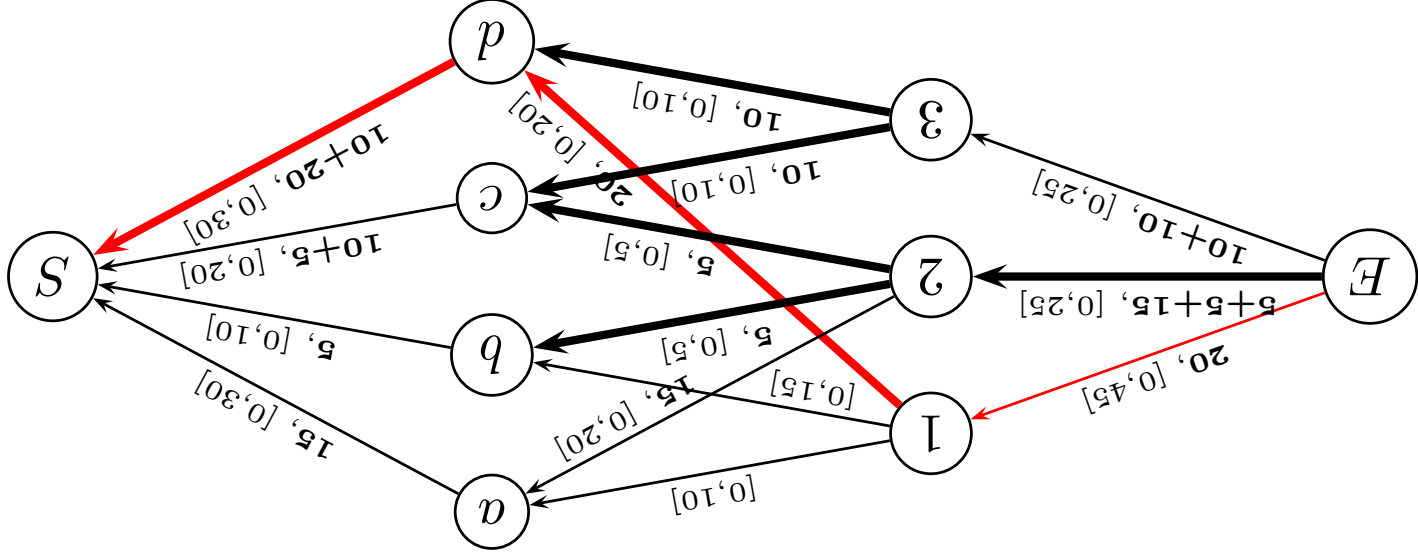
$E \rightarrow 3 \rightarrow d \rightarrow S : 10,$ $E \rightarrow 3 \rightarrow c \rightarrow S : 10$
 $E \rightarrow 2 \rightarrow c \rightarrow S : 5$
 $E \rightarrow 2 \rightarrow b \rightarrow S : 5$
 Total : 30

Construction d'un flot complet



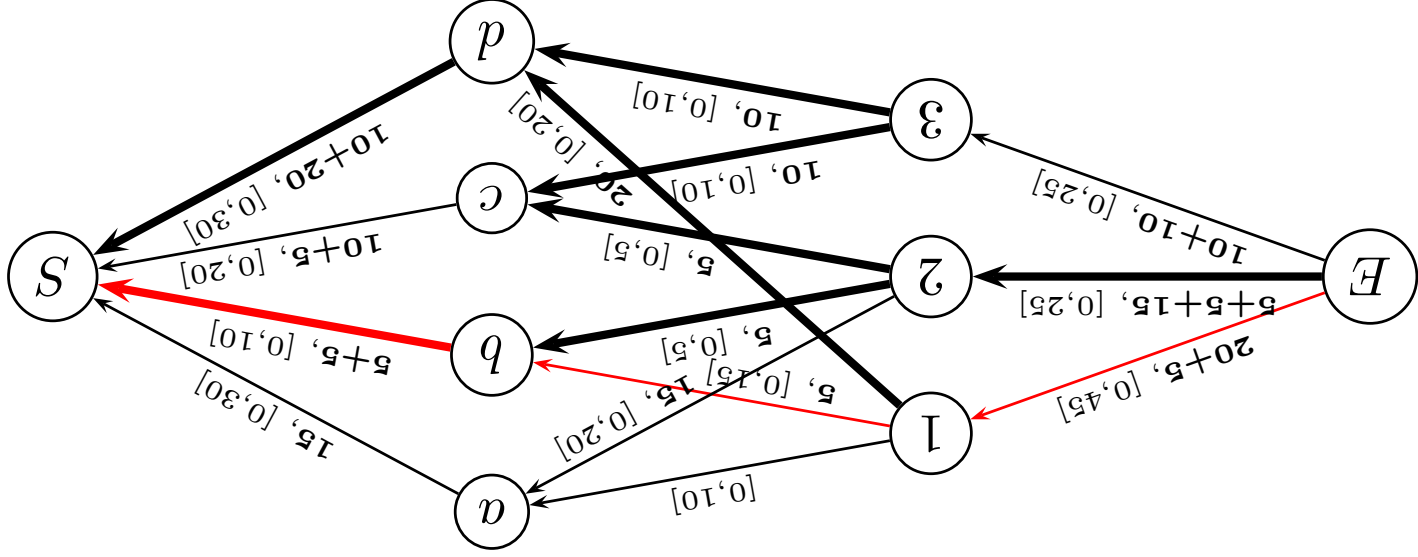
$E \rightarrow 3 \rightarrow d \rightarrow S : 10,$
 $E \rightarrow 2 \rightarrow c \rightarrow S : 5,$
 $E \rightarrow 2 \rightarrow a \rightarrow S : 15$
 Total : 45

Construction d'un flot complet



$E \rightarrow 3 \rightarrow d \rightarrow S : 10,$ $E \rightarrow 2 \rightarrow c \rightarrow S : 5,$
 $E \rightarrow 2 \rightarrow a \rightarrow S : 15$ $E \rightarrow 1 \rightarrow d \rightarrow S : 20$
 Total : 65

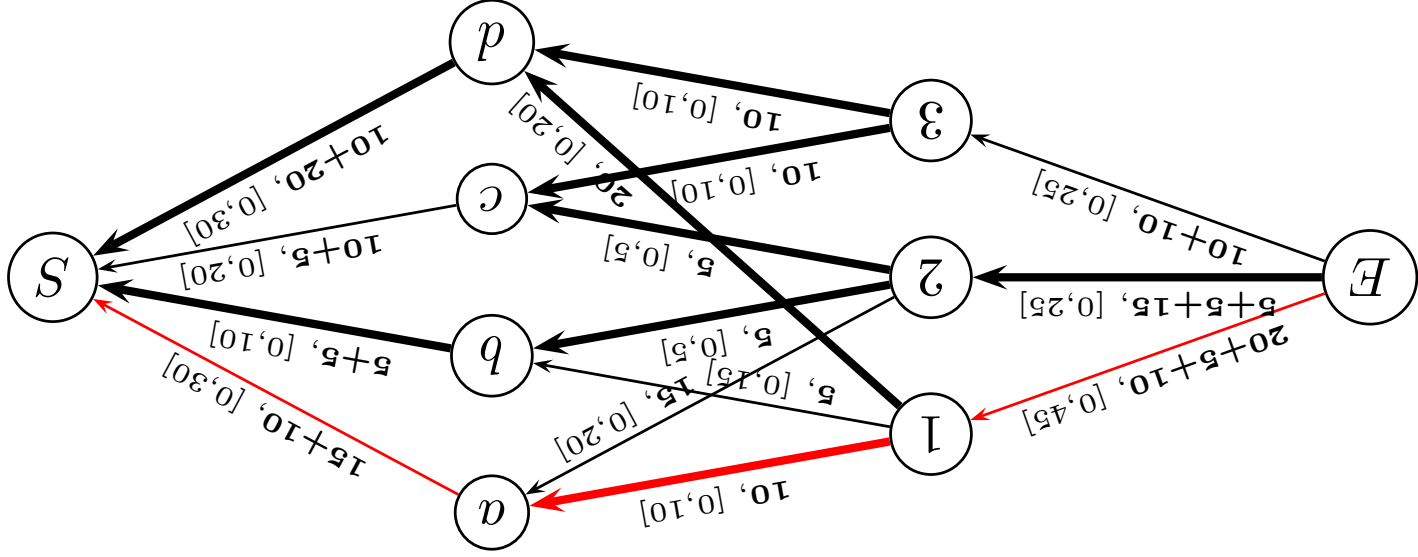
Construction d'un flot complet



$E \rightarrow 3 \rightarrow d \rightarrow S : 10,$
 $E \rightarrow 2 \rightarrow c \rightarrow S : 5,$
 $E \rightarrow 1 \rightarrow b \rightarrow S : 5$
 $E \rightarrow 2 \rightarrow a \rightarrow S : 15,$
 $E \rightarrow 1 \rightarrow d \rightarrow S : 20$
 $E \rightarrow 1 \rightarrow b \rightarrow S : 5$

Total : 70

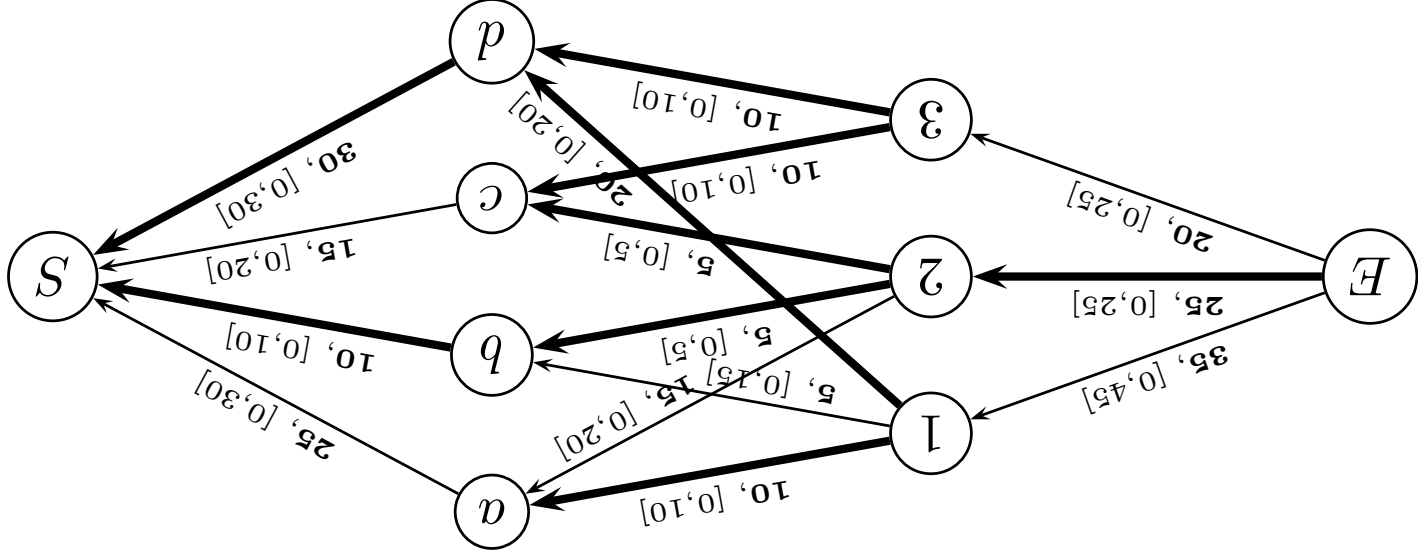
Construction d'un flot complet



- $E \rightarrow 3 \rightarrow d \rightarrow S : 10,$
- $E \rightarrow 3 \rightarrow c \rightarrow S : 10$
- $E \rightarrow 2 \rightarrow c \rightarrow S : 5,$
- $E \rightarrow 2 \rightarrow b \rightarrow S : 5$
- $E \rightarrow 2 \rightarrow a \rightarrow S : 15,$
- $E \rightarrow 1 \rightarrow d \rightarrow S : 20$
- $E \rightarrow 1 \rightarrow b \rightarrow S : 5$
- $E \rightarrow 1 \rightarrow a \rightarrow S : 10$

Total : 80

Flot complet



$E \rightarrow 3 \rightarrow d \rightarrow S : 10,$
 $E \rightarrow 2 \rightarrow c \rightarrow S : 5,$
 $E \rightarrow 2 \rightarrow a \rightarrow S : 15,$
 $E \rightarrow 1 \rightarrow b \rightarrow S : 5,$
 $E \rightarrow 1 \rightarrow d \rightarrow S : 20$
 $E \rightarrow 3 \rightarrow c \rightarrow S : 10$
 $E \rightarrow 2 \rightarrow b \rightarrow S : 5$
 $E \rightarrow 1 \rightarrow a \rightarrow S : 10$

Total : 80

3.3. Amélioration d'un flot réalisable

- L'algorithme de Ford-Fulkerson se décompose en deux étapes :
 - procédure de marquage + procédure de changement de flot.

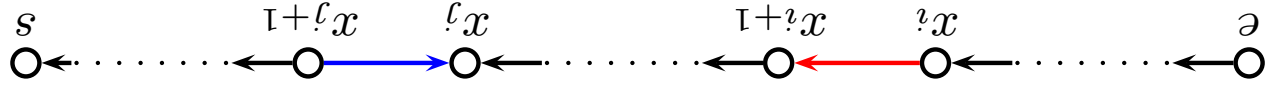
3.3.1. Procédure de marquage

```

début
  marquer [+] le sommet  $E$ 
  répéter
    - sélectionner un sommet marqué  $x$ ,
    - marquer [+] tout sommet  $y$  non marqué,
      extrémité terminale d'un arc  $(x, y)$  non saturé
    - marquer [-] tout sommet  $y$  non marqué,
      l'extrémité initiale d'un arc  $(y, x)$ 
      tel que  $\phi(y, x) < b(y, x)$ 
    jusqu'à (aucun sommet ne peut être marqué) ou
      ( $S$  marqué)
    si  $S$  n'est pas marqué
      alors le flot est maximum
    sinon le flot peut être amélioré
  fin si
fin
  
```

3.3.2. procédure de changement de flot

- Cette procédure ne s'applique que si S est marqué à l'issue de la procédure de marquage.
- Soit μ une chaîne de E à S .



- Soit μ_+ l'ensemble des arcs de la chaîne de type (x_i, x_{i+1}) "arcs avant".
- Soit μ_- l'ensemble des arcs de la chaîne de type (x_j, x_{j+1}) "arcs arriere".

- Les arcs de μ_+ sont tels qu'il est possible d'augmenter la valeur du flux associé d'une quantité $\varepsilon_+ \geq 0$.
- Les arcs de μ_- sont tels qu'il est possible de diminuer la valeur du flux associé d'une quantité $\varepsilon_- \geq 0$.

3.3.2. procédure de changement de flot

- Une chaîne μ est dite améliorante si $\varepsilon_+ > 0$ et $\varepsilon_- > 0$ (cf. §3.6 pour l'identification de telles chaînes).
- Si on augmente la valeur du flux sur les arcs de μ_+ et diminue la valeur du flux d'une même valeur sur les arcs de μ_- , la loi de Kirshhoff est préservée en chaque sommet,

Procédure de changement de flot ($\Phi \leftarrow \Phi'$) :

$$\begin{array}{l} \text{début} \\ \varepsilon_+ \rightarrow \min_{n \in \mu_+} \{k(n) - \phi(n)\} \\ \varepsilon_- \rightarrow \min_{n \in \mu_-} \{\phi(n) - b(n)\} \\ \varepsilon \rightarrow \min\{\varepsilon_+, \varepsilon_-\} \\ \text{fin} \end{array}$$

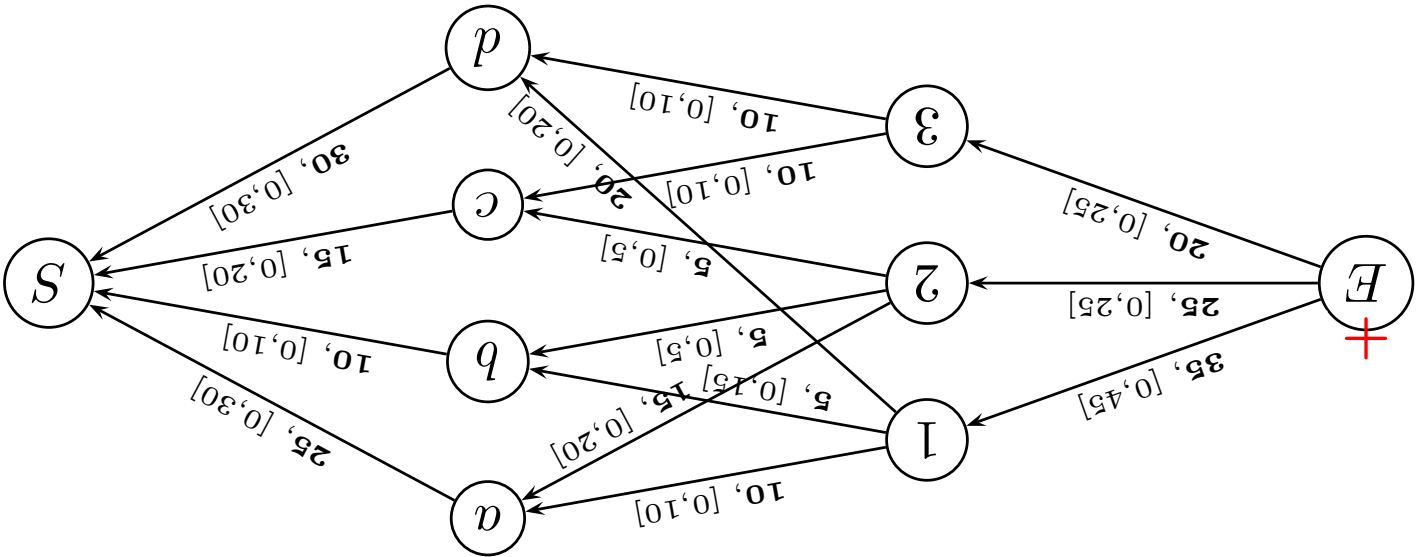
construire Φ' tel que

$$\begin{array}{l} \phi'_+(n) \rightarrow \phi(n) + \varepsilon, \forall n \in \mu_+ \\ \phi'_-(n) \rightarrow \phi(n) - \varepsilon, \forall n \in \mu_- \\ \phi'_i(n) \rightarrow \phi(n), \forall n \notin \mu \end{array}$$

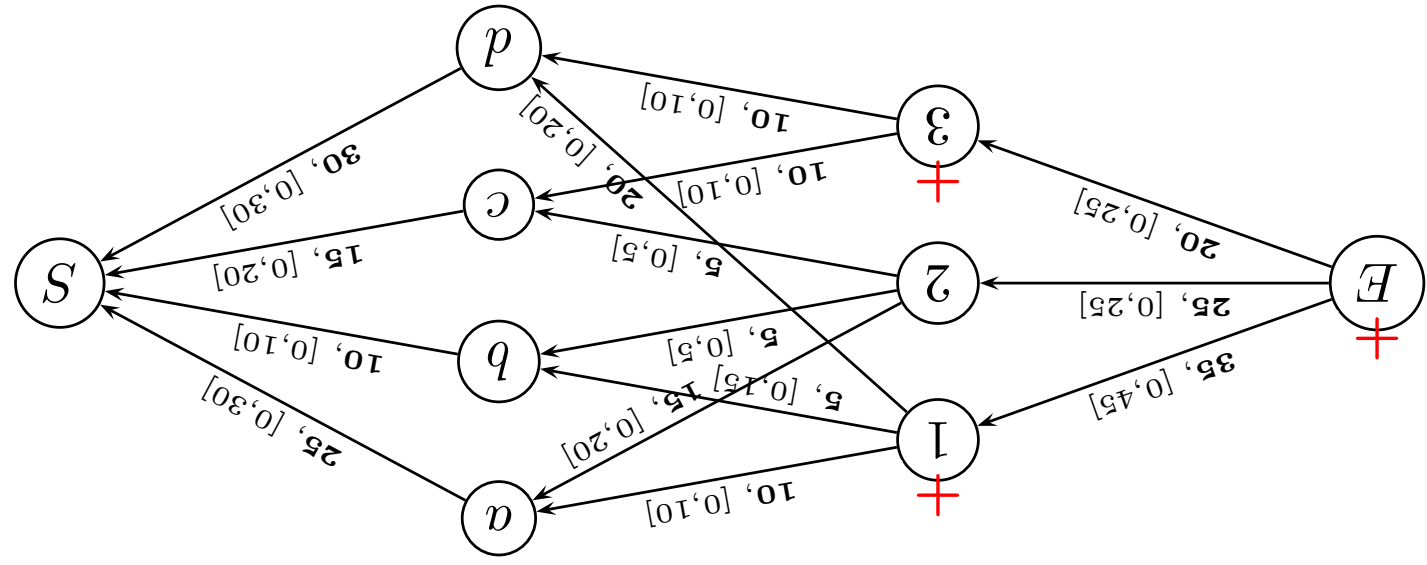
réappliquer la procédure de marquage

fin

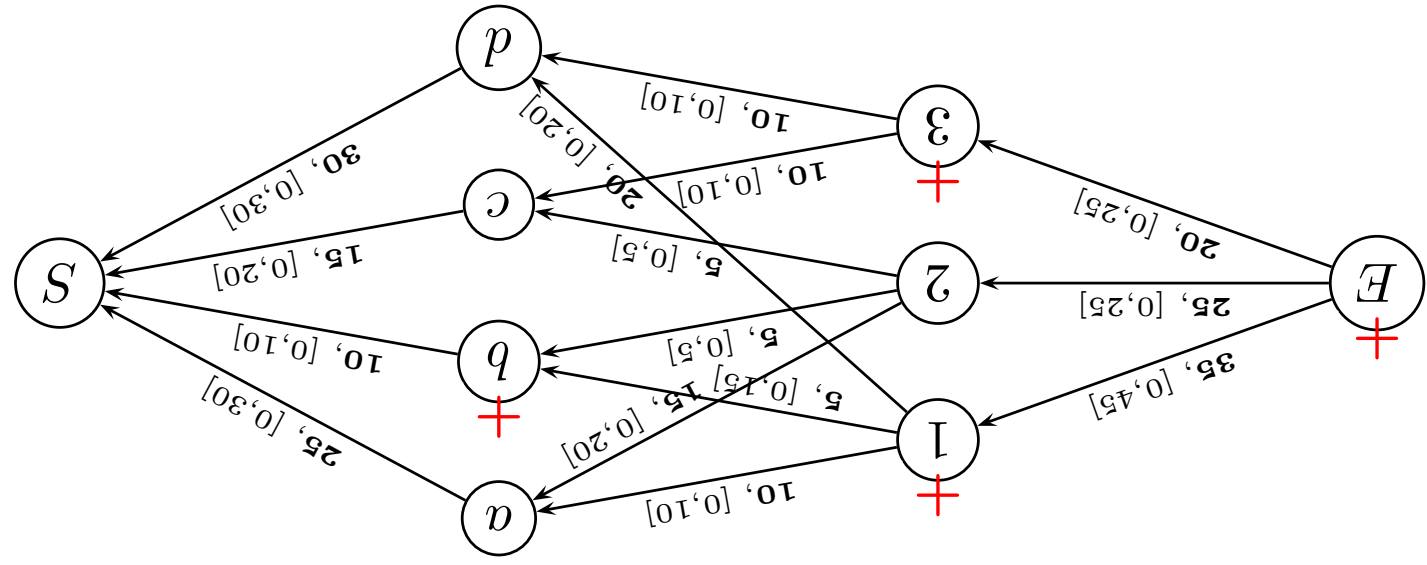
Marquage



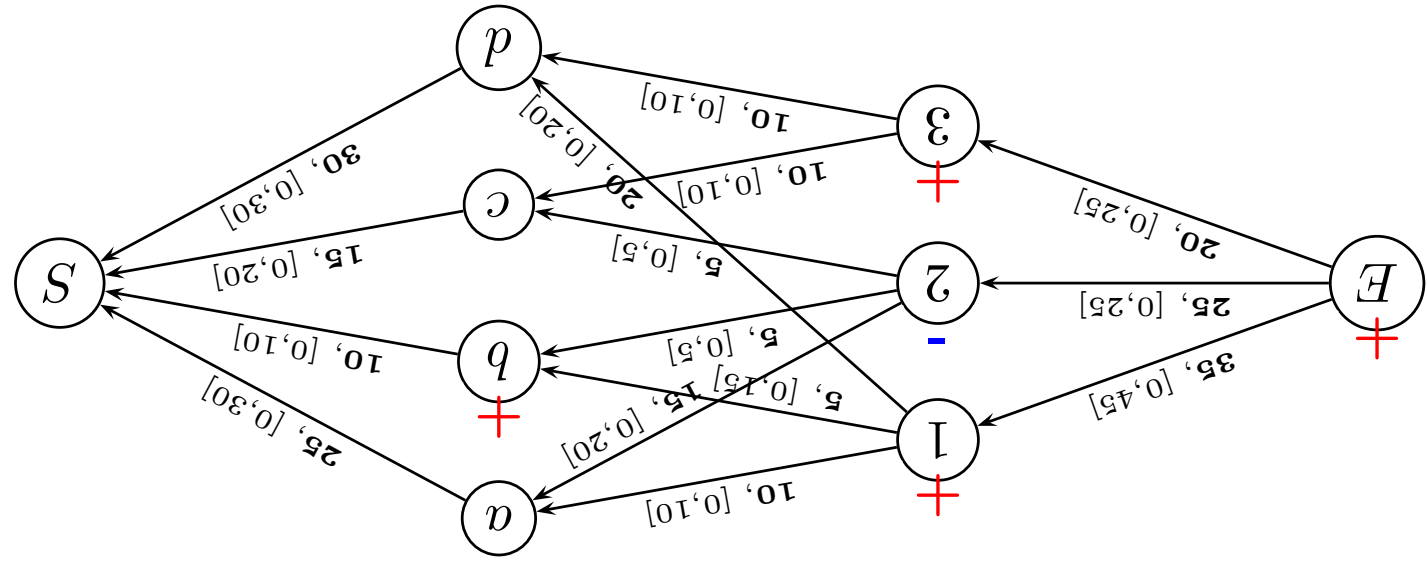
Marquage



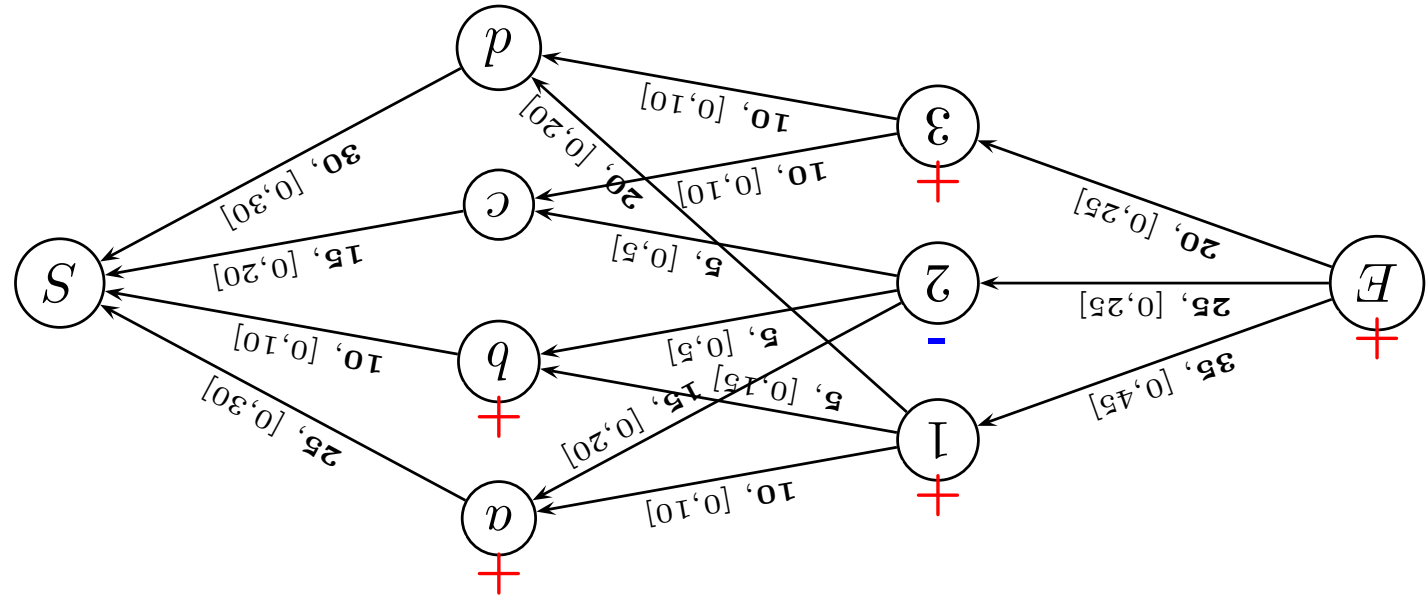
Marquage



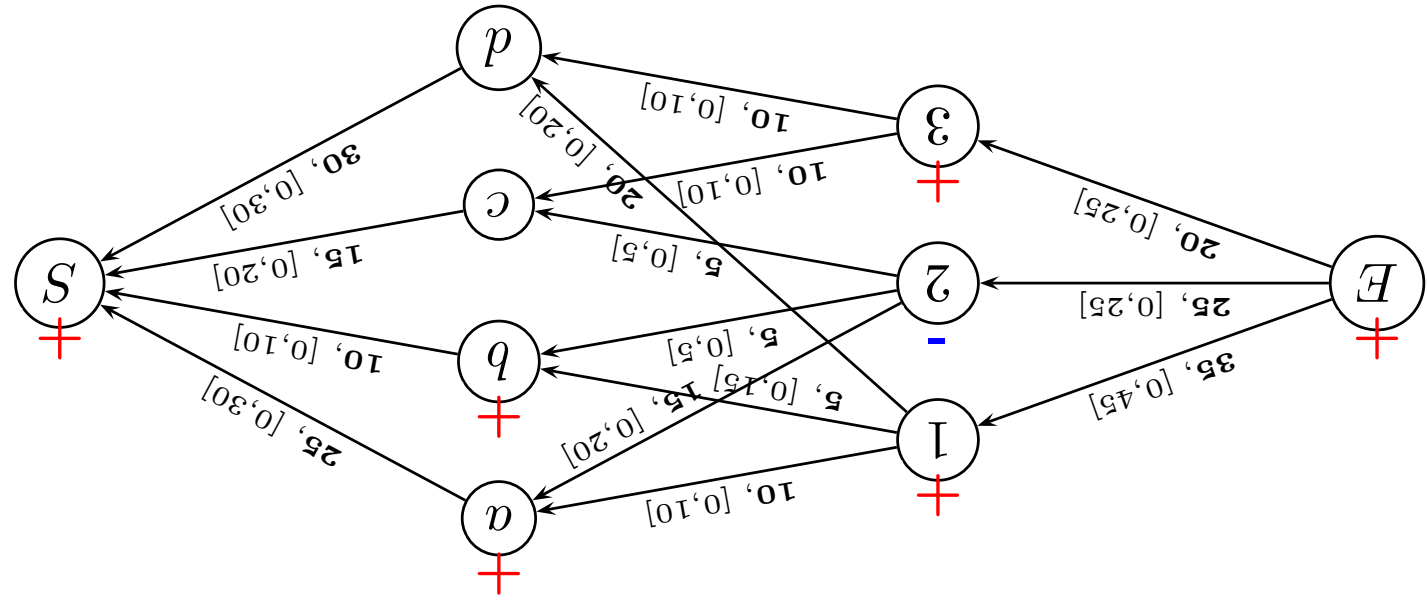
Marquage

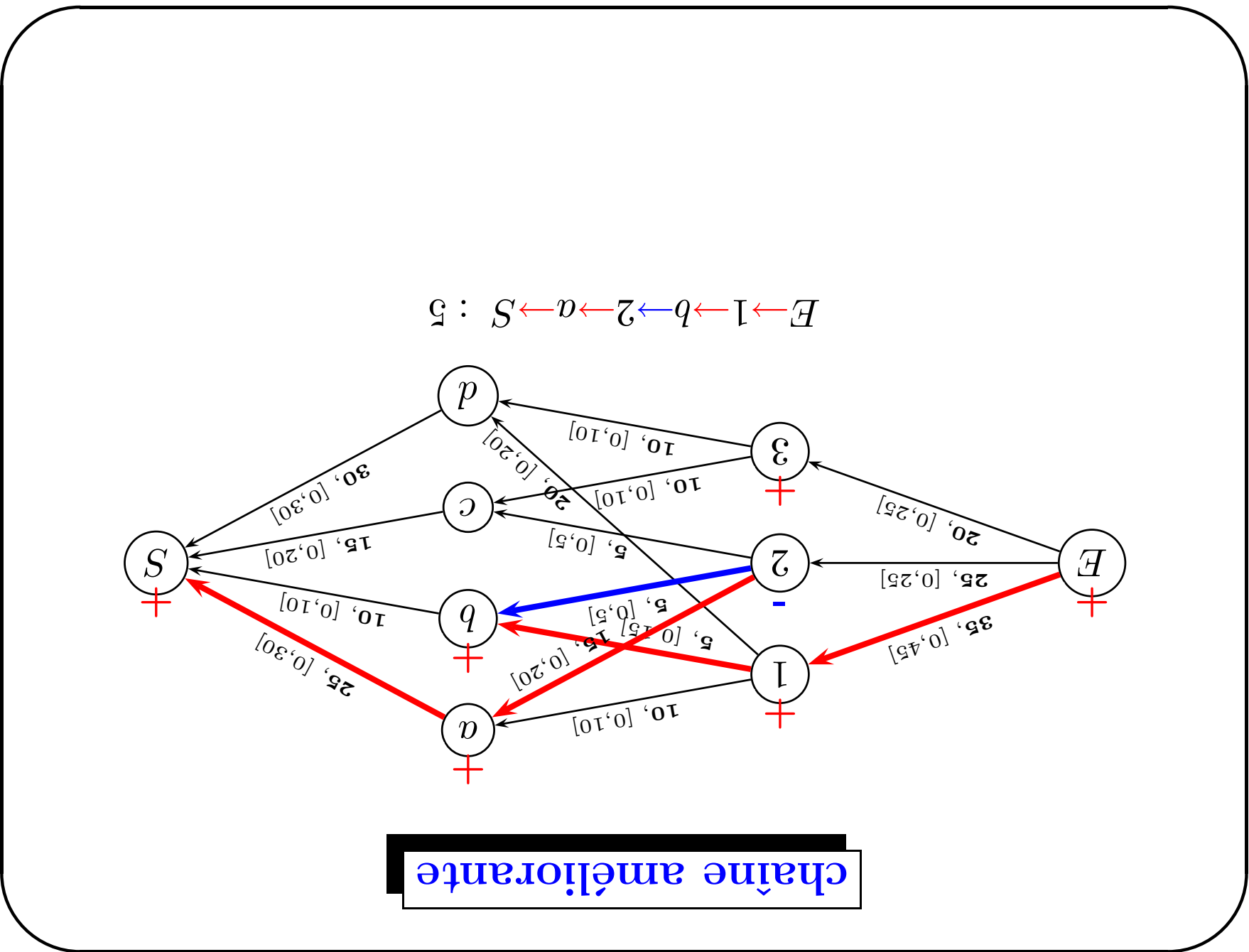


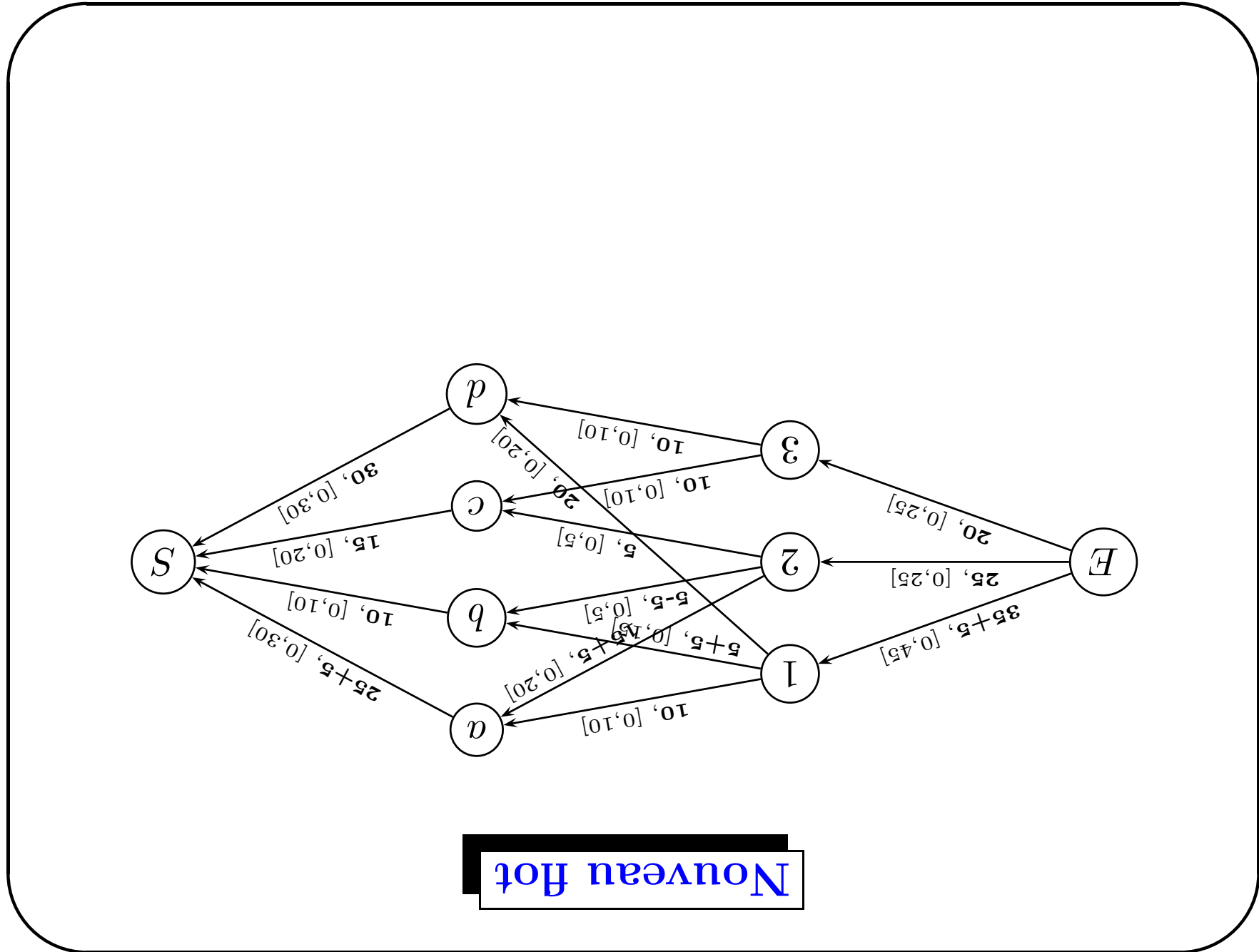
Marquage



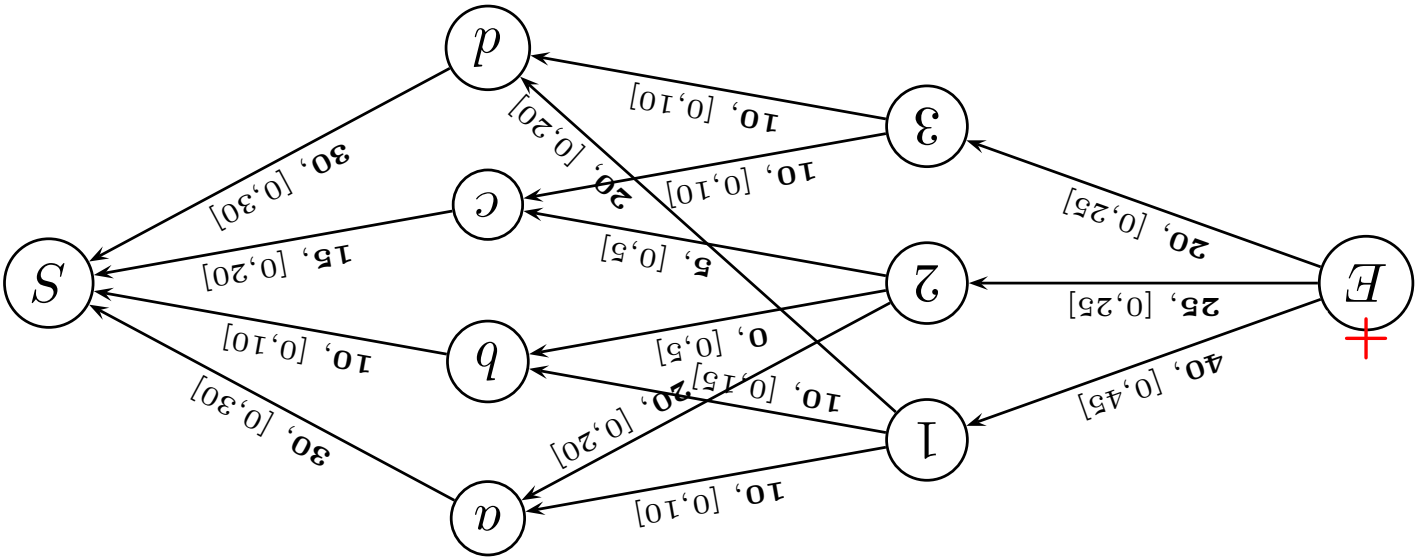
Marquage





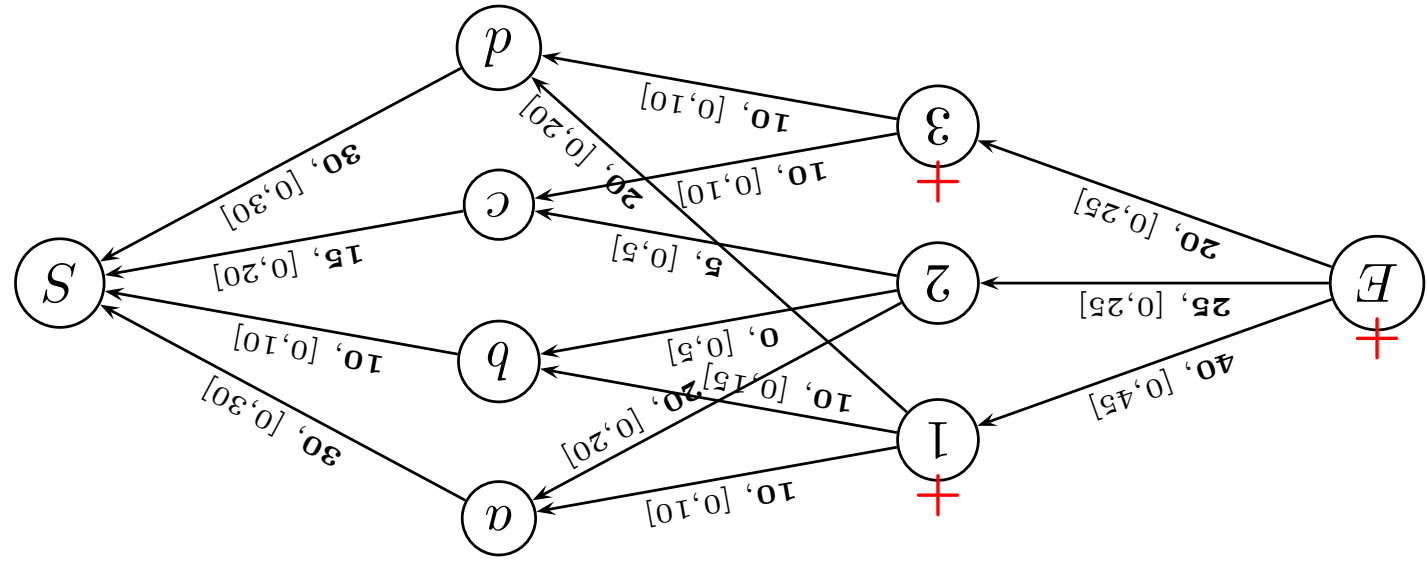


Nouveau flot

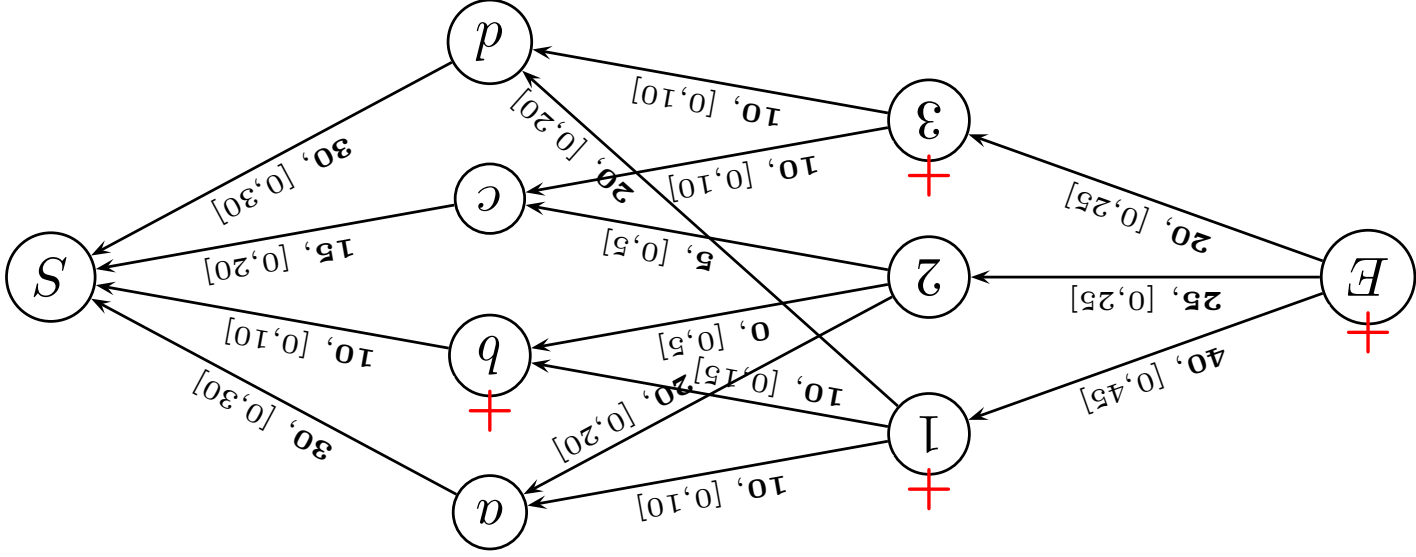


Marquage

Marguage

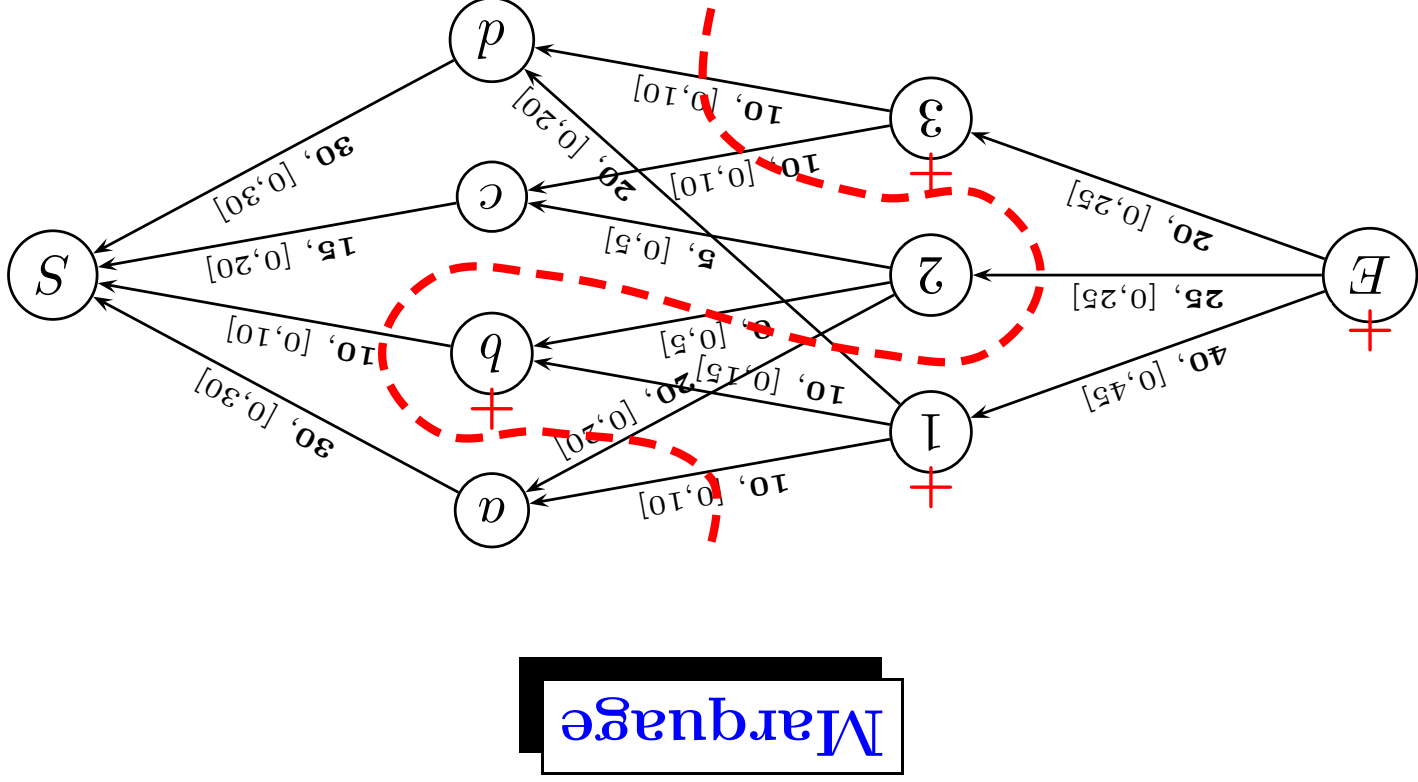


Marquage



On ne peut plus marquer de sommets, S n'est pas marqué

→ le flot est maximum.



On ne peut plus marquer de sommets,

S n'est pas marqué

→ le flot est maximum, $v(\Phi^*) = 85$.

La coupe de valeur minimale est engendrée par le sous-ensemble A des sommets marqués → $v(\omega(A^*)) = 85$.

3.4. Justification de l'algorithme

- Théorème (de Ford-Fulkerson) : La valeur d'un flot maximum est égale à la valeur d'une coupe minimum,

- Ce théorème traduit le fait que rechercher un flot maximum est équivalent à rechercher une coupe minimum,

- A la fin de l'algorithme de Ford-Fulkerson, soit A^* les sommets marqués ($H \in A^*$, $S \notin A^*$), la coupe $\omega(A^*)$ est de valeur

minimale,

- Lemme 1 : Soit Φ un flot défini sur un réseau de transport (de racine H et d'antiracine S), $\forall A \subset X$ t.q. $H \in A, S \notin A$, on a :

$$v(\Phi) = \sum_{n \in \omega^+(A)} \phi(n) - \sum_{n \in \omega^-(A)} \phi(n)$$

i.e., "la valeur d'un flot peut être déterminée sur toute coupe"

Preuve

- d'après la loi de Kirshhoff on a : $\sum_{n \in \omega_+^{(x)}} \phi(n) = \sum_{n \in \omega_-^{(x)}} \phi(n), \forall x \neq E$
- d'où : $\sum_{x \in A \setminus \{E\}} \sum_{n \in \omega_+^{(x)}} \phi(n) = \sum_{x \in A \setminus \{E\}} \sum_{n \in \omega_-^{(x)}} \phi(n)$
- C'est-à-dire : $\sum_{x \in A} \sum_{n \in \omega_+^{(x)}} \phi(n) - \sum_{n \in \omega_+^{(E)}} \phi(n) = \sum_{x \in A} \sum_{n \in \omega_-^{(x)}} \phi(n) - \sum_{n \in \omega_-^{(E)}} \phi(n)$,
- or, $\sum_{n \in \omega_+^{(E)}} \phi(n) = v(\Phi)$ et $\sum_{n \in \omega_-^{(E)}} \phi(n) = 0$
- donc,

$$\begin{aligned}
 v(\Phi) &= \sum_{x \in A} \sum_{n \in \omega_+^{(x)}} \phi(n) - \sum_{n \in \omega_+^{(E)}} \phi(n) \\
 &= \sum_{x \in A} \left(\sum_{y, z: y \in A, z \in A} \phi(y, z) + \sum_{y, z: y \notin A, z \notin A} \phi(y, z) \right) \\
 &= \sum_{n \in \omega_+^{(A)}} \phi(n) - \sum_{n \in \omega_-^{(A)}} \phi(n)
 \end{aligned}$$

□

Corollaire : $v(\Phi) = \sum_{n \in \omega^-(S)} \phi(n)$

Preuve : application du lemme 1 avec $A = \{S\}$

□

Lemme 2 (relation fondamentale flot-coupe) :

$\forall \Phi$ flot réalisable, $\forall A \subset X$ t.q. $E \in A, S \notin A$,

on a : $v(\Phi) \leq v(\omega(A))$

Preuve :

• d'après le lemme 1 on a :

$$v(\Phi) = \sum_{n \in \omega^+(A)} \phi(n) - \sum_{n \in \omega^-(A)} \phi(n),$$

• or Φ est réalisable donc $b(n) \leq \phi(n) \leq k(n)$, $\forall n \in U$,

• d'où $\sum_{n \in \omega^+(A)} \phi(n) \leq \sum_{n \in \omega^+(A)} k(n)$ et

$$-\sum_{n \in \omega^-(A)} \phi(n) \leq -\sum_{n \in \omega^-(A)} b(n)$$

• donc $v(\Phi) \leq \sum_{n \in \omega^+(A)} k(n) - \sum_{n \in \omega^-(A)} b(n)$

$$= v(\omega(A))$$

□

Théorème : Si \exists flot Φ^* et une coupe engendrée par A^* tels que $v(\Phi^*) = v(\omega(A^*))$, alors Φ^* est de valeur maximale et la coupe engendrée par A^* est de valeur minimale.

Preuve :

- d'après le lemme 2, $\forall \Phi'$ un flot réalisable, on a :
 $v(\Phi') \leq v(\omega(A^*))$, or $v(\omega(A^*)) = v(\Phi^*)$ donc $v(\Phi') \leq v(\Phi^*)$.
- d'après le lemme 2, $\forall A' \subset X$ t.q. $E \in A, S \notin A$, on a :
 $v(\omega(A')) \geq v(\Phi^*)$, or $v(\Phi^*) = v(\omega(A^*))$ donc
 $v(\omega(A')) \geq v(\omega(A^*))$
- Montrons qu'il existe un flot Φ^* et une coupe engendrée par A^* tels que $v(\Phi^*) = v(\omega(A^*))$,

- A la fin de l'algorithme de Ford-Fulkerson, on obtien un flot Φ_{ff} (réalisable par construction) et un ensemble de sommets marqués A_{ff} (avec $E \in A_{ff}$ et $S \notin A_{ff}$),

- on a :
 - $\forall u = (x, y) \in \omega_+(A_{ff}), u$ saturé (i.e. $\phi(u) = k(u)$), sinon on aurait pu marquer $+y$ et donc $u = (x, y) \notin \omega_+(A_{ff})$,
 - $\forall u = (x, y) \in \omega_-(A_{ff}), \phi(u) = b(u)$, sinon on aurait pu marquer $-x$ et donc $u = (x, y) \notin \omega_-(A_{ff})$,

- d'après le lemme 1, on a :

$$v(\Phi_{ff}) = \sum_{u \in \omega_+(A_{ff})} \phi(u) - \sum_{u \in \omega_-(A_{ff})} \phi(u) = \sum_{u \in \omega_+(A_{ff})} k(u) - \sum_{u \in \omega_-(A_{ff})} b(u) = v(\omega(A_{ff}))$$

□

- l'algorithme de Ford-Fulkerson fournit donc de déterminer un flot de valeur maximum et une coupe de valeur minimum,
- La preuve du théorème de Ford-Fulkerson est fournie par l'algorithme lui-même.

3.5. Convergence et complexité

- Supposons que toutes les capacités et bornes $k(u)$ et $b(u)$, $\forall u \in U$ sont entières (non restrictif d'un point de vue de l'implémentation),

- Montrons que l'algorithme converge en un nombre fini d'étapes,
 - partant du flot nul Φ^0 à chaque étape de l'algorithme (prise en compte d'une chaîne améliorante), le flot est augmenté d'un valeur entière ? 0

- or $v(\Phi^*) \leq v(\omega(A))$, $\forall A \subset X$ t.q. $E \in A, S \notin A \Rightarrow$ il existe un nombre fini de chaînes améliorantes,
- $b(u)$ et $k(u)$ sont entier \Rightarrow l'algorithme converge en un nombre fini d'itérations.

Complexité de l'algorithme de Ford-Fulkerson,

- nombre d'itérations $\leq v(\Phi^*) = v(\omega^+({E}))$,
- à chaque itération, on examine tous les sommets (excepté E), c'est-à-dire $n - 1$ sommets

- examiner un sommet consiste à effectuer un nombre

d'opération proportionnel au degré du sommet (pour tester si on peut prolonger une chaîne améliorante),

- le nombre d'opérations à effectuer à chaque itération est donc borné par la somme des degrés des sommets, *i.e.*, $2m$,

- la complexité est donc de $O(m.v(\Phi^*))$ ou $O(m.v(\omega({E})))$,

- or $\omega({E})$ est majoré par $n.k_{max}$, où $k_{max} = \max_{u \in U} k(u)$, donc la complexité est de $O(m.n.k_{max})$

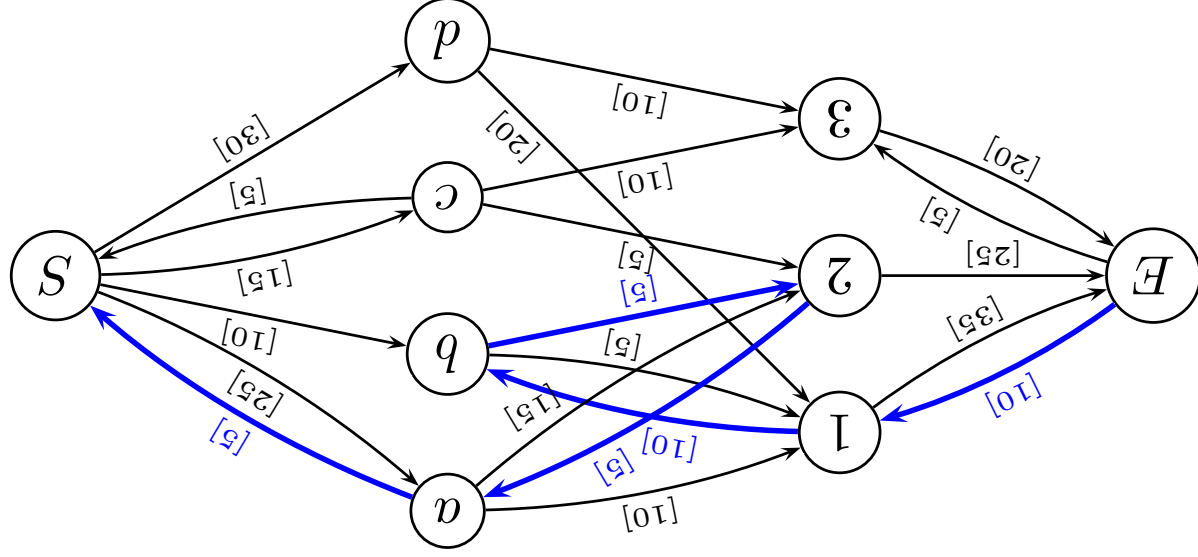
3.6. Graphe d'écart

On peut associer à un réseau de transport $G = (X, U)$ antisymétrique et un flot Φ , un graphe $G^e(\Phi) = (X, U^e(\Phi))$ appelé graphe d'écart. $U^e(\Phi)$ est défini par :

- à tout arc $(x, y) \in U$ tel que $b(n) > \phi(n) > k(n)$, on associe :
 - un arc $(x, y) \in U^e(\Phi)$ t.q. $k_e(x, y) = k(n) - \phi(n)$ et $c_e(x, y) = c(n)$,
 - un arc $(y, x) \in U^e(\Phi)$ t.q. $k_e(y, x) = b(n) - \phi(n)$ et $c_e(y, x) = -c(n)$,
- à tout arc $(x, y) \in U$ tel que $\phi(n) = b(n) = k(n)$, on associe un arc $(x, y) \in U^e(\Phi)$ t.q. $k_e(x, y) = k(n) - \phi(n) = 0$ et $c_e(x, y) = c(n)$,

3.6. Graphe d'écart

- Dans le graphe $G^e(\Phi) = (X, U^e(\Phi))$, tout chemin de S à E représente une chaîne améliorante ,
- Considérons le flot Φ correspondant au flot complet déterminé au §3.2. le graphe $G^e(\Phi)$ est le suivant. On retrouve la chaîne améliorante $E1b2aS$



Changement de flot avec $G^e(\Phi)$

- Soit μ^e un chemin de E à S dans $G^e(\Phi)$ et μ la chaîne correspondante (concept non-orienté) dans G ,

- Soit $\delta = \text{Min}_{n \in \mu^e} k^e(n)$,

- On peut obtenir un flot réalisable Φ' en faisant varier les flux sur les arcs de G comme suit :

– $\forall u \in \mu$ t.q. l'arc correspondant dans μ^e a la même orientation : $\phi'(u) = \phi(u) + \varepsilon$ ($\varepsilon \leq \delta$),

– $\forall u \in \mu$ pour lequel l'arc correspondant dans μ^e a l'orientation inverse : $\phi'(u) = \phi(u) - \varepsilon$ ($\varepsilon \leq \delta$),

– $\forall u \notin \mu, \phi'(u) = \phi(u)$

Changement de flot avec $G^\varepsilon(\Phi)$

- Φ est un flot (loi de Kirshhoff préservée) réalisable ($\varepsilon \leq \delta$),
- En appliquant la procédure à un chemin de E à S , on augmente la valeur du flot Φ de ε ,

- En appliquant la procédure à un circuit, on ne modifie pas la valeur du flot mais on a : $c(\Phi') = c(\Phi) + \sum_{n \in \mu^\varepsilon} c^\varepsilon(n) \cdot \varepsilon$,

- Exemple : circuit $1b2a1$, $\delta = \min\{10, 5, 10, 10\} = 5$

- ajouter 5 à $\phi(1, b)$,
- retirer 5 à $\phi(2, b)$,
- ajouter 5 à $\phi(2, a)$,
- retirer 5 à $\phi(1, a)$,

4. Flot de coût minimum

- Soit un réseau de transport $G = (X, U)$ sur lequel sont associées à chaque arc $n \in U$, $b(n)$ (borne), $k(n)$ (capacité) et $c(n)$ (coût unitaire),

- Le problème consiste à déterminer parmi les flots réalisables de valeur v , un flot de coût minimum,

- Formulation en Programmation Linéaire :

$$\begin{aligned} \min \quad & \sum_{n \in U} c(n) \cdot \phi(n) \\ \text{s.c.} \quad & \sum_{n \in \omega^-(x)} \phi(n) = \sum_{n \in \omega^+(x)} \phi(n), \forall x \neq E, S \\ & b(n) \leq \phi(n) \leq k(n), \forall n \in U \\ & \sum_{n \in \omega^+(E)} c(n) \cdot \phi(n) = v \\ & \phi(n) \geq 0, \forall n \in U \end{aligned}$$

- Les variables du PL sont $\phi(u)$, $\forall u \in U$, les deux premières contraintes assurent que le flot est réalisable,
- La dernière contrainte assure que les flots considérés sont de valeur v ,
- Cas particulier : flot maximum à coût minimum on pose $v = v(\Phi_*)$ dans la dernière contrainte.

- Théorème : Φ , un flot de valeur v , est de coût minimum ssi $G^e(\Phi)$ ne contient pas de circuit de valeur négative au sens des coûts,
- Algorithme de construction d'un flot de valeur v et de coût minimum :

début
 construire un flot Φ de valeur v de coût k
 répéter
 - construire $G^e(\Phi)$,
 - si $\exists \mu$ un circuit de valeur $k_\mu > 0$ dans $G^e(\Phi)$,
 alors construire Φ' en appliquant la
 procédure de changement de flot
 sur μ avec $\varepsilon = \delta$
 Finsi
 - le flot Φ' est de valeur $v' = v$ et de
 coût $k' = k - \delta \cdot k_\mu$
 jusqu'à $\nexists \mu$ circuit de valeur négative dans $G^e(\Phi)$
 fin

- On peut appliquer différents algorithmes pour la détection de circuits de valeur négative (Bellman-Kalaba, Floyd,...).
- *Problème* : A partir d'un flot Φ de valeur v ($v < v(\Phi^*)$) et de coût minimum, comment déterminer un flot Φ' de valeur $v' > v$ et de coût minimum ?

- Théorème :

Soit Φ un flot de valeur v ($v < v(\Phi^*)$) et de coût minimum, soit μ^e un chemin de coût minimum de E à S dans $G^e(\Phi)$. On obtient un flot Φ' de valeur $v' > v$ et de coût minimum en appliquant la procédure de changement de flot sur μ^e .

- Algorithme de construction d'un flot Φ' de valeur $v' > v$ et de coût minimum à partir d'un flot Φ de valeur v et de coût minimum :

- début
- soit Φ un flot de coût minimum parmi les flots de valeur v
 - construire $G^e(\Phi)$,
 - choisir dans $G^e(\Phi)$ un chemin μ^e de H à S minimal au sens des coûts
 - construire Φ' en appliquant la procédure de changement de flot en utilisant avec $\varepsilon = \delta$
- fin
- *Problème* : pour appliquer l'algorithme, il faut disposer d'un flot de coût minimum. On peut soit appliquer l'algorithme précédent, soit considérer le flot nul.