Flots canalisés

Soit le réseau $R' = (X, A, b, c)$ une extension du réseau $R = (X, A, c)$ étudié précédemment où $b : A \to \mathbb{R} \bigcup \{-\infty\}$, $c : A \to \mathbb{R} \bigcup \{+\infty\}$ et $\forall a,\ b(a) \leq c(a)$.

Définition

Un flot dans le réseau R' est une application $f : A \to \mathbb{R}^+$ telle que :

- $\forall a,\ b(a) \leq f(a) \leq c(a)$
- $\forall w \in X, \sum_{\vec{uw} \in A} f(uw) = \sum_{\vec{wv} \in A} f(wv)$ — conservation du flot

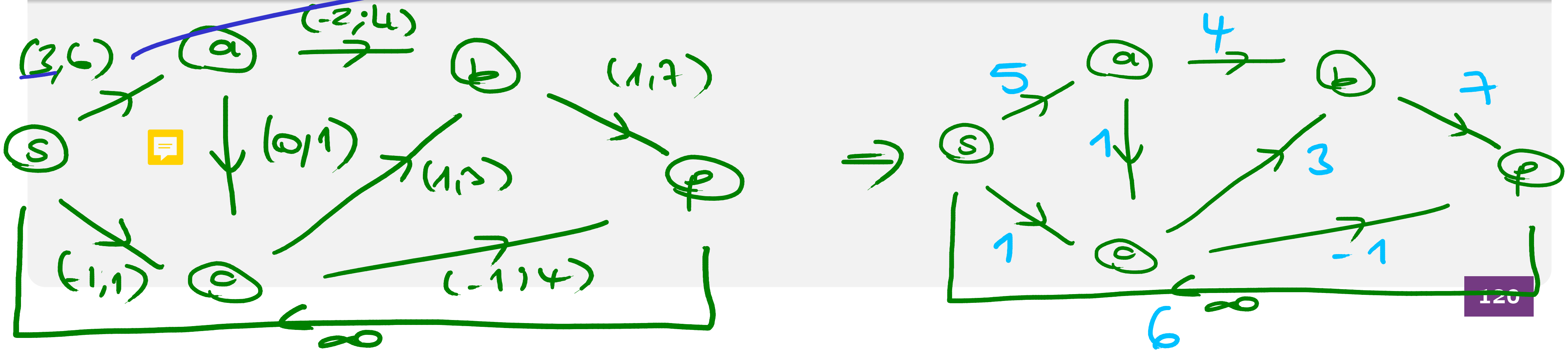
Ce flot est dit **canalisé**.

Remarque : le problème du flot maximum correspond à celui-ci avec $b(a) = 0$, pour tout arc a.

$(b(a), c(a))$

entre S et P, on a au moins 3 u de flot et au plus 6 u

Exemple



Flots canalisés : amélioration d'un flot

La recherche d'un flot canalisé maximum peut être effectuée par une extension très simple de l'algorithme de Ford-Fulkerson en modifiant la procédure de marquage inverse.

Tout simplement prendre en compte la borne min de b(a) dans le calcul lors d'un marquage inverse. On ne peut repousser qu'au maximum la difference entre le flot et b(a).
Si 2<3<5, on ne peut pas repousser 3 mais uniquement 1

Marquage inverse modifié

Si un sommet x est marqué ($\delta(x)$) et
s'il existe y non marqué tel que $f(yx) > b(yx)$
Alors :

$$\delta(y) \leftarrow min[\delta(x), f(yx) - b(yx)] \qquad \text{(et } y \text{ est marqué)}$$

Ainsi l'algorithme modifié de Ford-Fulkerson permet d'améliorer un flot existant. Reste le problème de la détermination d'un flot de départ, car contrairement au problème classique, le flot $\Phi = 0$ n'est pas toujours une solution réalisable.

Flots canalisés : recherche d'un flot initial

Théorème de Hoffman

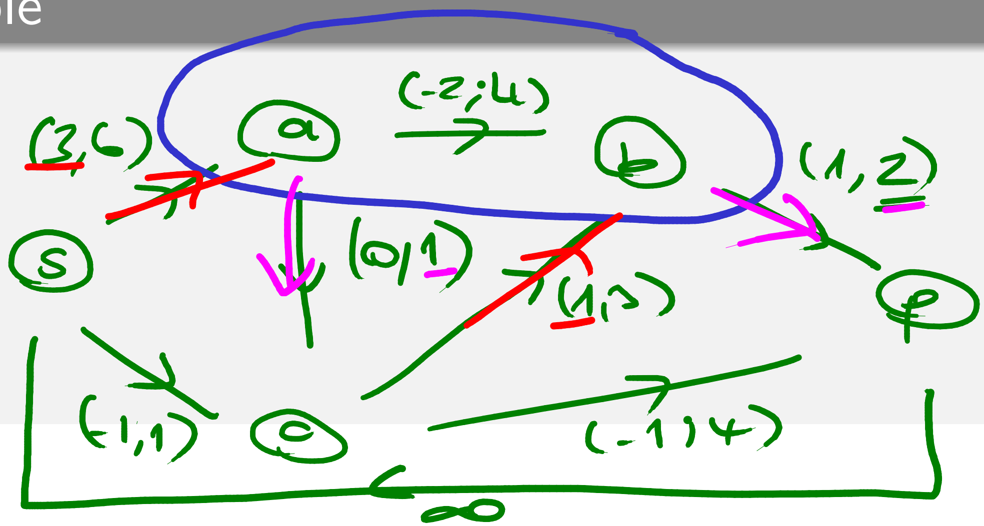
Il existe un flot canalisé sur $R = (X, A, b, c)$
si et seulement si pour tout cocycle $\omega(Y)$ du graphe (X, A) $(Y \subset X)$,
on a :

$$\underbrace{\sum_{a \in \omega^-(Y)} b(a)}_{\text{min}} \leq \overbrace{\sum_{a \in \omega^+(Y)} c(a)}^{\text{max}}$$

$\forall Y, \Sigma \leq \Sigma \Leftrightarrow$ flot canalisé

$\hookrightarrow \exists Y, \Sigma > \Sigma \Rightarrow \nexists$ flot canalisé

Exemple



But : Mq il n'existe pas de flot canalisé.

$Y = \{a, b\}$

$\sum_{a \in \omega^-(Y)} b(a) = 3 + 1 = 4$

$>$

$\sum_{a \in \omega^+(Y)} c(a) = 1 + 2 = 3$

$\Rightarrow \nexists$ flot canalisé

La recherche d'un flot initial peut se faire en modifiant le graphe et en appliquant itérativement la procédure de recherche d'un flot canalisé maximum. Nous ne détaillerons pas ici cette méthode un peu complexe.

Application : Problème des cases admissibles (1)

On se donne un tableau rectangulaire $m \times n$ (m lignes, n colonnes) dont certaines cases sont interdites (les autres sont dites "admissibles"). On se donne également $m + n$ entiers positifs ou nuls d_1, d_2, ... d_m et b_1, b_2, ... b_n.
Le problème consiste à associer à chaque case admissible un nombre entier de telle sorte que :

- La somme des nombres affectés aux cases d'une ligne i soit inférieure ou égale à d_i $(i = 1, 2, ...m)$.
- La somme des nombres affectés aux cases d'une colonne j soit inférieure ou égale à b_j $(j = 1, 2, ...n)$.
- La somme des nombres affectés à l'ensemble des cases soit maximum.

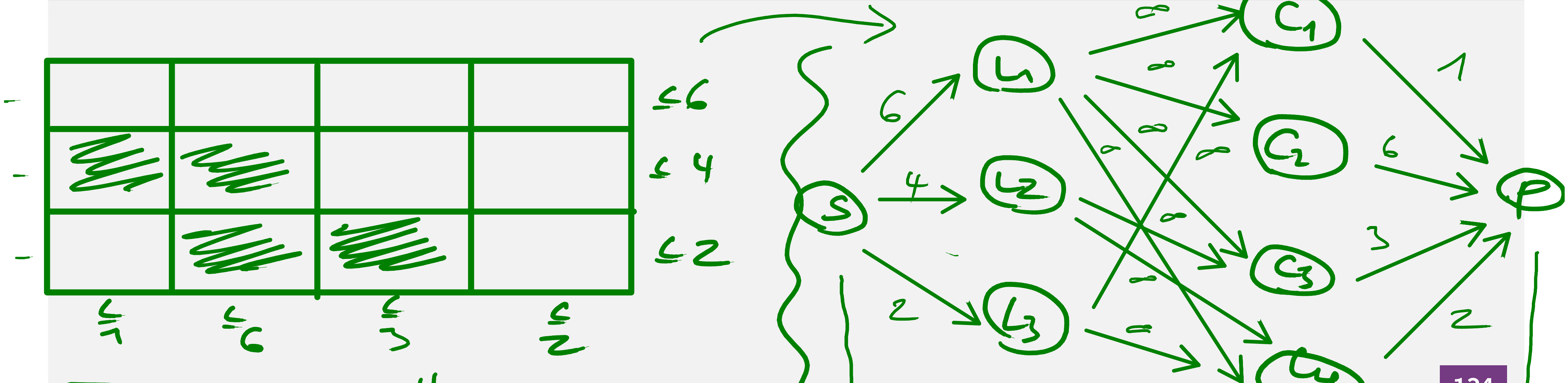
Application : Problème des cases admissibles (2)

Ce problème revient **à rechercher le flot maximum** dans le réseau $R = (X, A, c)$ défini de la façon suivante :

- $X = L \bigcup C \bigcup \{s, p\}$ où $L = \{l_1, l_2, \ldots l_m\}$ et $C = \{c_1, c_2, \ldots c_n\}$.
- $A = A_1 \bigcup A_2 \bigcup A_3 \bigcup \{a_r\}$ où
  - $A_1 = \{(sl_i) \mid i = 1, 2, \ldots m\}$
  - $A_2 = \{(c_j p) \mid j = 1, 2, \ldots n\}$
  - $A_3 = \{(l_i c_j) \mid (i, j)$ est une case admissible $\}$
  - $a_r = (ps)$
- et $c : A \to \mathbb{R}$ telle que

$$c(a) = \begin{cases} +\infty & si \quad a = a_r \text{ ou } a \in A_3 \\ d_i & si \quad a = (sl_i) \in A_1 \\ b_j & si \quad a = (c_j p) \in A_2 \end{cases}$$

Exemple



Σ colonne 1 ≤ 1 || Σ ligne 1 ≤ 6

Recherche d'un flot maximum de coût minimal (1)

Soit le réseau $R = (X, A, c, k)$ une extension du réseau $R = (X, A, c)$, où $k : A \to \mathbb{R}^+$ tel que $k(a)$ représente le coût unitaire de l'arc a.

Objectif : rechercher dans ce réseau le flot maximal de coût minimal

Ce problème bi-objectif (flot maximal, coût minimal) est difficile à résoudre. En effet que privilégier ? Le flot ? Le coût ?

Ici on recherche **le flot de coût minimal parmi les flots de valeur maximale**.

Le coût (valeur) d'un flot est calculé par :

$$v(f) = \sum_{(xy) \in A} k(xy) \cdot f(xy)$$

Recherche d'un flot maximum de coût minimal (2)

Extension de la notion de graphe d'écart aux réseaux avec coûts

Pour un réseau $R = (X, A, c, k)$ et un flot f réalisable dans ce réseau. Au couple (R, f) on associe un **graphe d'écart** $R^e(f) = (X, A^e, c^e, k^e)$ où :

- R^e a le même ensemble de sommets que R,
- À tout arc $(xy) \in A$, de capacité $c(xy)$ et de flux $f(xy)$, on associe dans $R^e(f)$:
  - un arc (xy)
    - de capacité $c^e(xy) = c(xy) - f(xy)$
    - de coût $k^e(xy) = k(xy)$

    si l'arc n'est pas saturé
  - un arc (yx)
    - de capacité $c^e(yx) = f(xy)$
    - de coût $k^e(yx) = -k(xy)$

    si le flux passant par (xy) n'est pas nul

Algorithme de Roy (= Roi)

Soit $R = (X, A, c, k)$. Cet algorithme recherche le flot maximum de coût minimal, en recherchant itérativement le chemin de coût minimal dans le graphe d'écart courant.

Structure

`Initialiser` f `à 0`

`Répéter`

    `Construire` $R^e(f) = (X, A^e, c^e, k^e)$ algo de Dijkstra

    `Recherche` Γ`, le chemin de coût minimal de` s `à` p `dans` $R^e(f)$

flot max // capacité max

    `Si` Γ `existe Alors`

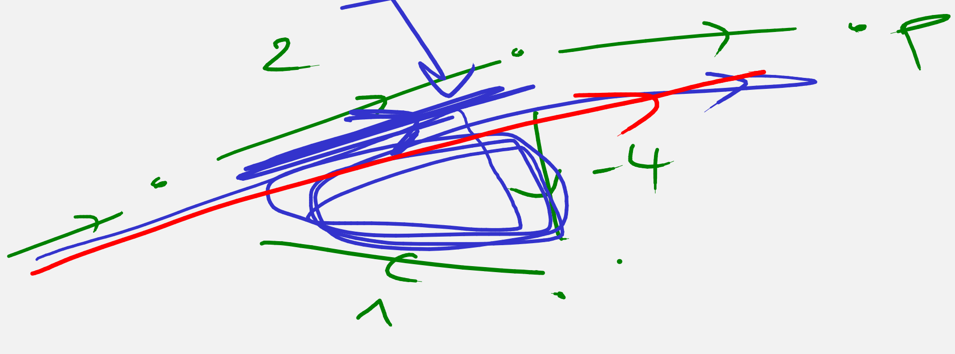
        $\epsilon := \min_{(xy)\in\Gamma}(c^e(xy))$

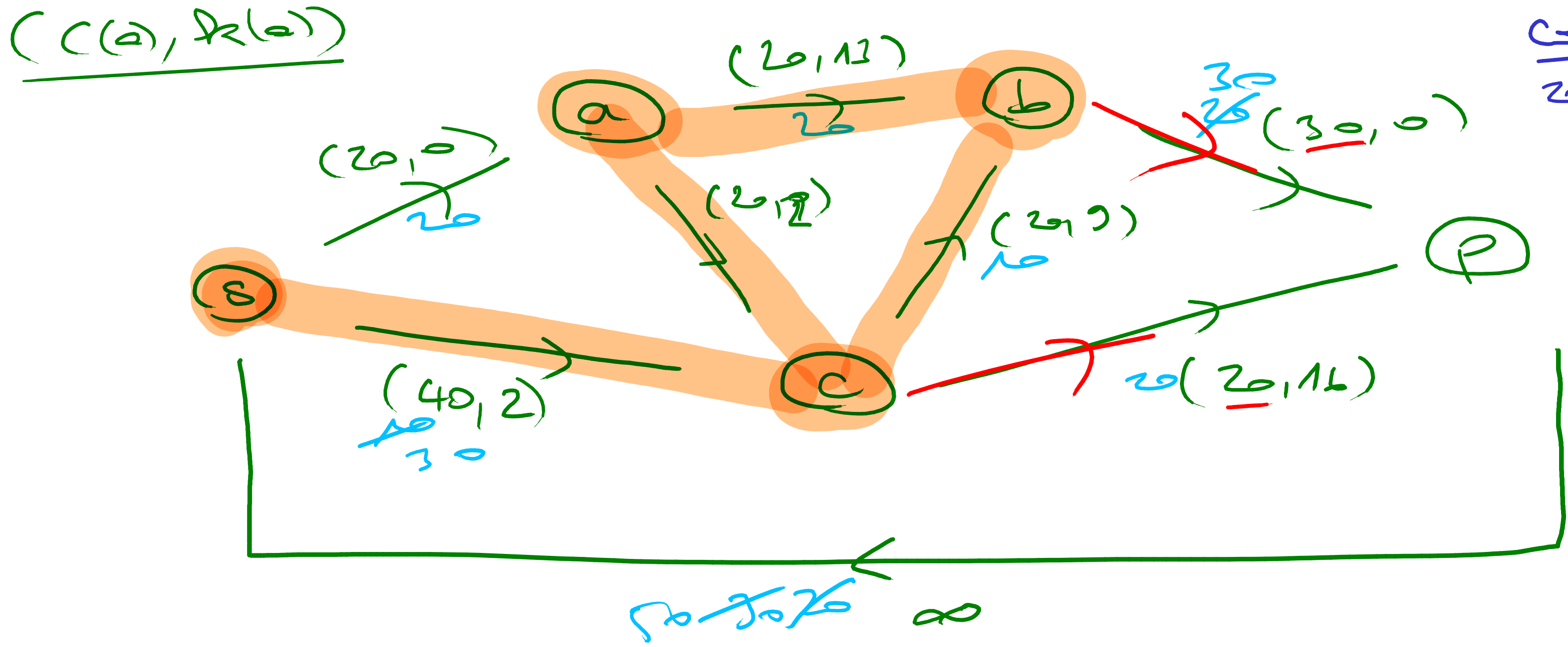
        $\forall (xy) \in \Gamma,\ f(xy) = f(xy) + \epsilon$

    `FinSi`

`Jusqu'à` Γ `n'existe pas`

`Calculer le coût du flot`





Coût $\phi = 50$

$20\times 0 + 20\times 13 + 30\times 0 + 10\times 9 + 30\times 2 + 20\times 16 + 0\times 0 = 730$

Algo de FF à partir du flot $\phi = 0$

E_1 : $\mu = \{s, a, b, p\}$ $\quad \delta = 20 \Rightarrow \phi = 20$

E_2 : $\mu = \{s, c, b, p\}$ $\quad \delta = 10 \Rightarrow \phi = 30$

E_3 : $\mu = \{s, c, p\}$ $\quad \delta = 20 \Rightarrow \boxed{\phi = 50}$

E_4 : $Y = \{s, a, b, c\}$ $\Rightarrow \bar{Y} = \{p\}$

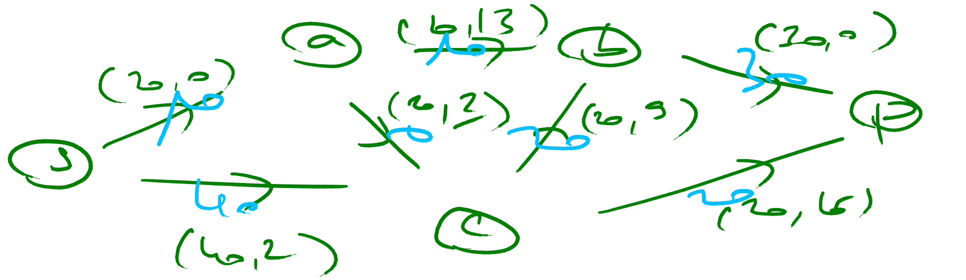
$\mathcal{C} = \{(b,p), (c,p)\}$

$c(\mathcal{C}) = 30 + 20 = 50$ $\Rightarrow \boxed{\phi^* = 50}$

Algorithme de Roy

Déroulement sur un exemple

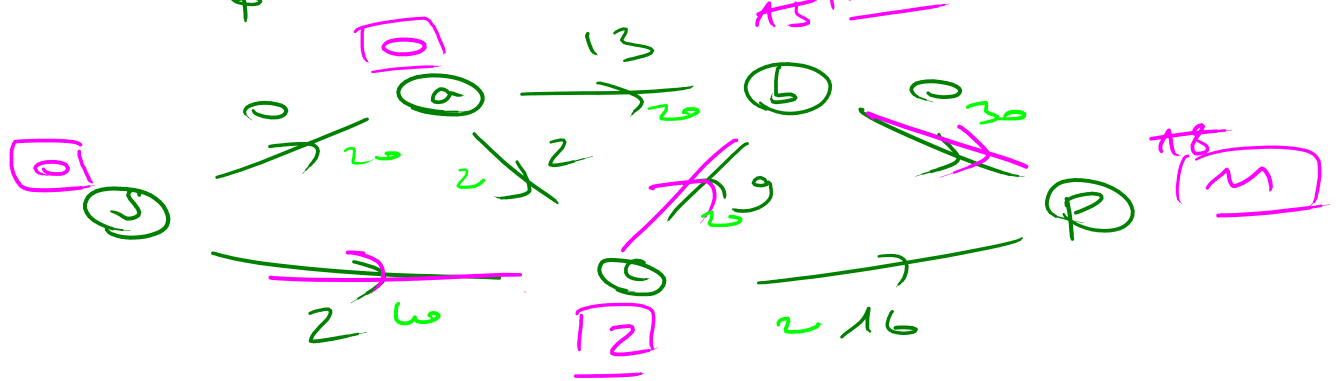
1



R et $\phi = 0$

2

$E_1 \quad R^e_\phi = R$



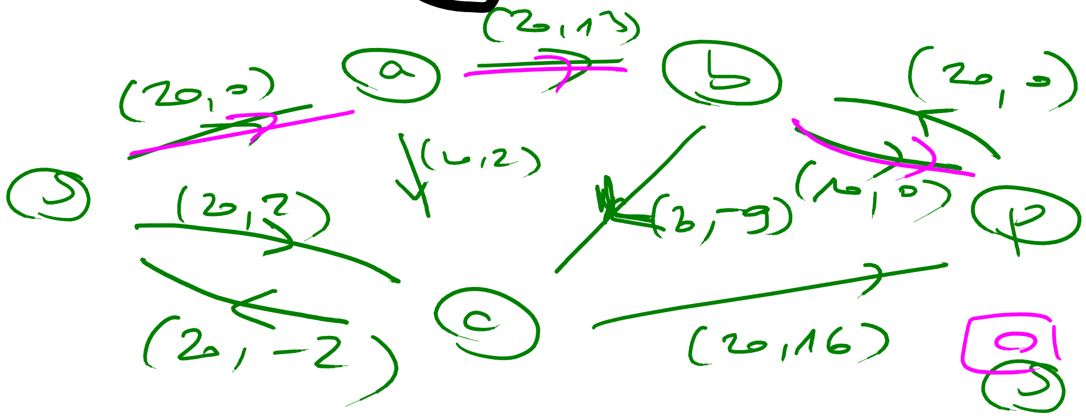
$\Gamma = \{s, c, b, p\}$

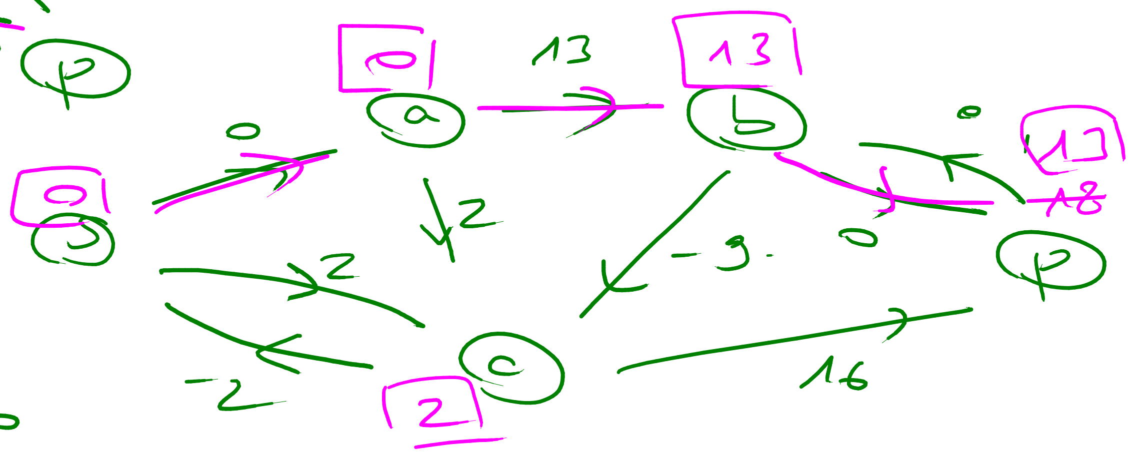
$\delta = 20 \Rightarrow \phi = 20$

$E_2 \quad R^e_\phi$

3

<- ce qui est passé
-> ce qui peut passer



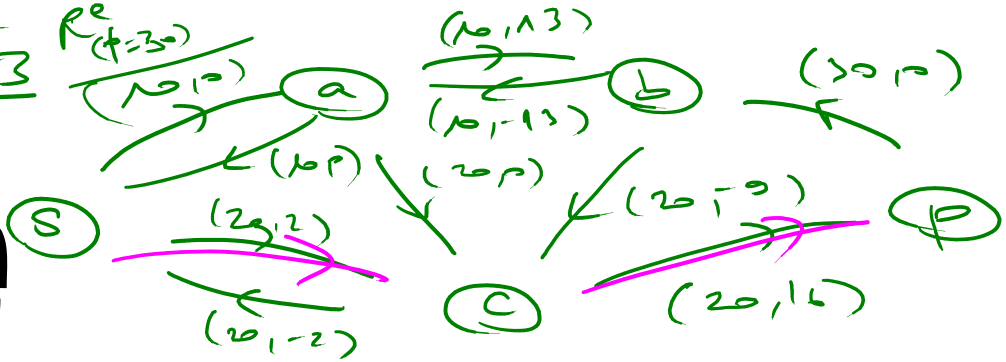


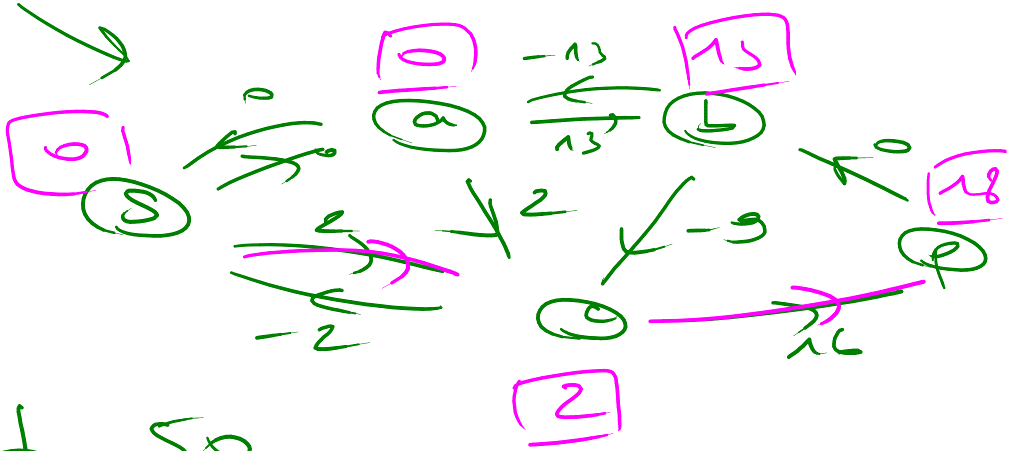
$\Gamma = \{s, a, b, p\}$

$\delta = 10 \Rightarrow \phi = 30$

Ez3 $R^e_{(\phi=30)}$

4



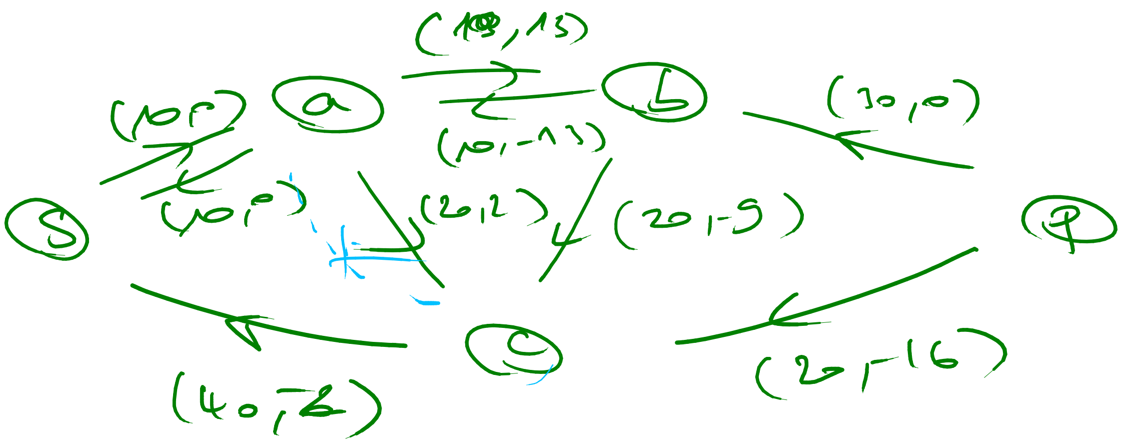


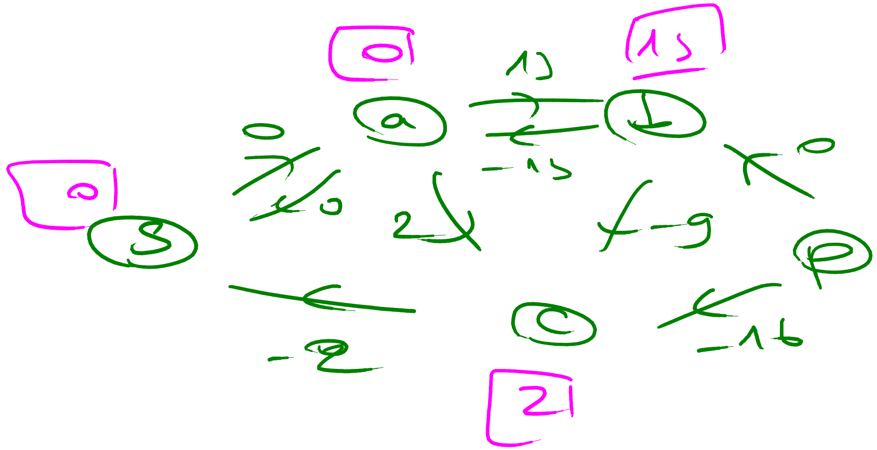
$\Gamma = \{s, c, t\}$

$\delta = 20 \Rightarrow \underline{\phi = 50}$

5

Ez4 $R^e_{(\phi=50)}$





→ pas de chemin de s à p

⇒ STOP

$\phi = 50$ – Coût : $10\times0 + 10\times13 + 30\times0 + 40\times2 + 20\times9 + 20\times16 + 0\times2 = 710$