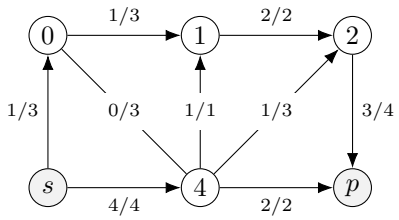
INFO601 : algorithmique et graphes
TD 5 : flot maximal

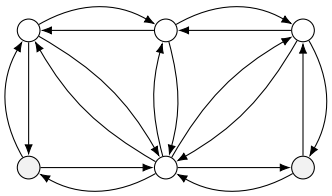
Pierre Hyvernat, Gérald Cavallini
Laboratoire de mathématiques de l'université de Savoie
bâtiment Chablais, bureau 17, poste : 94 22
email : Pierre.Hyvernat@univ-smb.fr
www : http://www.lama.univ-smb.fr/~hyvernat/

Exercice 1 : Exemples

Question 1. On considère un réseau non-orienté avec le flot suivant :



- Vérifiez attentivement qu'il s'agit effectivement d'un flot.
- Qu'elle est sa valeur ?
- Construisez le graphe des flots résiduels en donnant la valeur de *toutes* les arêtes :



Question 2. En utilisant le réseau et le flot de la question précédente, cherchez un chemin augmentant pour mettre le flot à jour. Continuez à appliquer l'algorithme de Ford-Fulkerson pour trouver un flot maximal. Quelle est sa valeur ?

Exercice 2 : Représentation

Question 1.

On suppose que le réseau initial et le flot sont donnés par des listes d'adjacence `C` et `F`. Écrivez les fonctions `incr_flow(F, s1, s2, f)` et `get_residual_flow(C, F, s1, s2)` qui

- incrémente le flot de l'arête `s1`→`s2` de la valeur `f`.
- calcule le flot résiduel de l'arête `s1`→`s2` (et renvoie `0` s'il n'y a pas d'arête de `s1` à `s2`).

Question 2. Pour éviter de parcourir les listes d'adjacence pour la mise à jours, on modifie les listes d'adjacence du flot `F` de la manière suivante. `F[s]` contient la liste des `arc` sortants de `s`, où un `arc` "`a`" contient :

- `a.target` : le but de l'arc,
- `a.capacity` : la capacité de cet arc,
- `a.flow` : la valeur du flot sur cet arc,
- `a.opposite` : référence à l'arc opposé. (Par exemple, l'indice de l'arc opposé (de `a.target` vers `s`) dans la liste `F[a.target]`.)

Récrivez la fonction `get_residual_flow(C, F, a)`, qui prend un `arc` en argument.

Question 3. Pour accéder aux arcs d'un chemin augmentant, il faut modifier le parcours pour qu'il renvoie un tableau `Pred` avec des `arc` plutôt que de simples prédécesseurs. Écrivez le parcours (en largeur) qui cherche un chemin dans le graphe des flots résiduels et renvoie le tableau `Pred` correspondant.

Question 4. En C, la fonction `incr_flow(F, a, f)` qui doit modifier l'arc `a` pourrait prendre un pointeur vers l'arc `a`. En Python, si les `arc` sont des objets mutables, on peut également modifier un `arc` directement (c'est un pointeur). Si les `arc` ne sont pas mutables, il faut modifier la case `F[o][idx_a]` contenant `a`.

Comment peut-on récupérer l'origine `o` d'un arc `a` et son indice dans `F[o]` *uniquement à partir de* `a`, et en temps constant ?

Écrivez la fonction `incr_flow(F, a, f)` correspondante qui met à jours l'arc `a` *et son opposé* dans ce cas.

Exercice 3 : compréhension

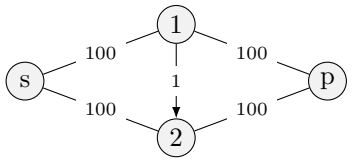
Question 1. Donnez un exemple de réseau et de flot maximal avec un flot strictement positif *entrant* sur la source (un "feedback").

Question 2. Est-il possible de trouver un réseau où tous les flots maximaux ont un flot strictement positif entrant sur la source ?

Question 3. Comment peut on adapter le problème du flot maximum lorsqu'il y a plusieurs sources et/ou plusieurs puits ?

Exercice 4 : Bien choisir les chemins augmentants

Question 1. Donnez un flot maximal (et sa valeur) pour le réseau suivant.



(Remarque : l'arête $1 \to 2$ est orientée. La capacité $2 \to 1$ vaut 0.)

Question 2. Montrez que l'algorithme de Edmonds-Karp (algorithme de Ford-Fulkerson avec un parcours en largeur) termine en 2 étapes sur ce flot maximal.

Question 3. Si on choisit les chemins de manière particulièrement malchanceuse, quel est le nombre d'étapes pour obtenir ce flot ?

Est-ce que cette situation pourrait arriver en pratique ?