

FEUILLE 4 : FLOT MAXIMAL DANS UN GRAPHE

Exercice 1. *Parcours dans un graphe orienté*

Le but de cet exercice est d'étudier deux façons différentes de parcourir un graphe orienté.

1) Parcours en profondeur (DFS, Depth First Search) avec une pile.

Avec ce parcours, on cherche à explorer le sommet le plus éloigné du sommet de départ, à la condition qu'il soit le successeur d'un sommet qui a déjà été exploré. Dans le cas du parcours en profondeur, on utilise une pile pour l'exploration des sommets (de type LIFO, Last In, First Out c'est-à-dire : "dernier entré, premier sorti"). A partir d'un sommet courant, on empile et on marque le premier successeur non encore marqué. Le sommet courant est ensuite enlevé de la pile et on utilise le sommet suivant de la pile pour continuer. On procède ainsi jusqu'à ce que la pile soit vide. L'algorithme générique s'écrit :

Initialement, tous les sommets sont non marqués et la pile est vide.

Tant qu'il existe¹ un sommet S non marqué,

 insérer S au sommet de la pile. Marquer S .

Tant que la pile n'est pas vide

 Soit Q sommet situé en haut de la pile.

S'il existe¹ un sommet P non marqué, successeur de Q

Alors insérer P en haut de la pile. Marquer P .

Sinon supprimer Q de la pile.

Fin Tant que

Fin Tant que

2) Parcours en largeur (BFS, Breath First Search) avec une file.

Dans ce type de parcours, on essaie toujours d'explorer le sommet le plus proche du sommet de départ et qui n'a pas déjà été visité. A partir d'un sommet courant, on liste et on marque tous ses successeurs non encore marqués. Le sommet courant est ensuite enlevé de la liste et on utilise le premier sommet de la liste pour continuer. On utilise une structure de file d'attente (de type FIFO, First in, first out c'est-à-dire "premier entré, premier sorti"). L'algorithme générique s'écrit :

Initialement, tous les sommets sont non marqués et la file est vide.

Tant qu'il existe un sommet S non marqué,

 insérer S dans la file. Marquer S .

Tant que la file n'est pas vide

 - Supprimer de la file le sommet Q situé en tête de file.

 - Insérer successivement dans la file tous les sommets P non marqués, successeurs de Q .

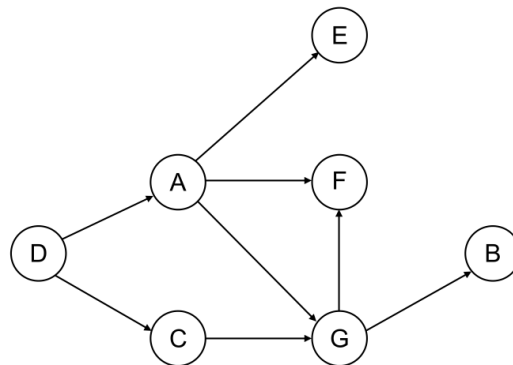
 - Marquer les sommets P .

Fin Tant que

Fin Tant que

1. S'il y a plusieurs sommets concernés, on prend le premier dans l'ordre lexicographique.

1. En partant du sommet D , parcourir le graphe ci-dessous en utilisant les deux méthodes différentes proposées. Pour chaque parcours, vous afficherez un tableau représentant l'état de la pile/file à chaque itération et vous indiquerez l'ordre de parcours des sommets.



2. Ecrire une version récursive de l'algorithme de parcours en profondeur.

3. *Parcours profondeur DFS et tri topologique.*

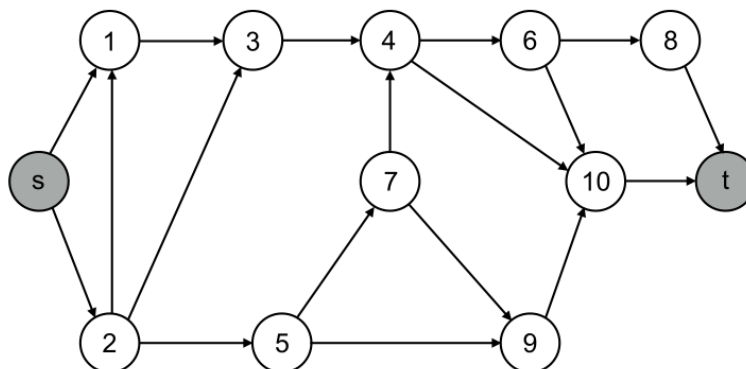
Le tri topologique d'un graphe *sans cycle*² consiste à ordonner les numéros des sommets de telle sorte que chaque sommet du graphe est toujours visité avant ses successeurs. Dans le parcours en profondeur DFS, en sauvegardant successivement les numéros des sommets défilés on obtient un tri topologique des sommets en ordre inverse.

- (a) A partir du parcours en profondeur DFS réalisé à la question 1, donner un tri topologique du graphe ci-dessus.
- (b) Modifier l'algorithme de parcours en profondeur DFS pour réaliser un tri topologique des sommets d'un graphe sans cycle.

4. *Parcours largeur BFS et distance minimale.*

La distance entre deux sommets d'un graphe orienté est définie comme le nombre d'arêtes du chemin reliant les deux sommets. On considère un graphe orienté tel que tous les sommets sont accessibles à partir d'un sommet initial s par un chemin. Le parcours en largeur (BFS) permet de calculer la distance minimale de tous les sommets au sommet initial. Pour calculer la distance minimale, on introduit la fonction L telle que $L(s) = 0$ pour le sommet s initial et on fait $L(P) = L(Q) + 1$ si on rencontre (et marque) le successeur non-marqué P de Q .

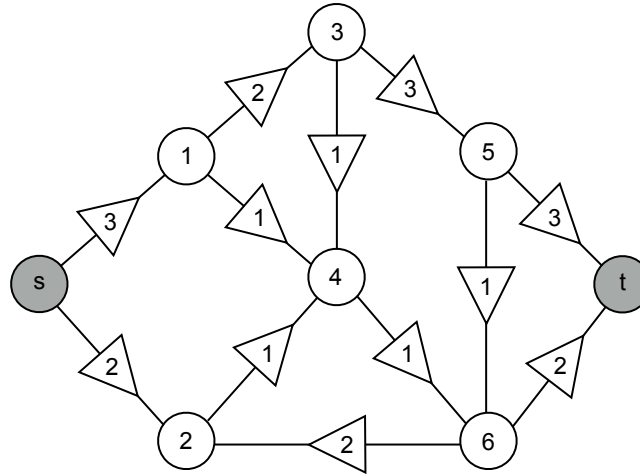
- (a) Modifier l'algorithme du parcours en largeur (BFS) pour calculer les distances minimales de chacun des sommets du graphe au sommet initial s .
- (b) Déterminer en effectuant un parcours largeur (BFS) la distance minimale de s à t pour le graphe suivant.



2. un cycle dans un graphe orienté est un chemin dont les deux sommets extrémités sont identiques

Exercice 2. *Flot maximal dans un graphe*

Calculer le flot maximal dans le graphe valué suivant, à l'aide de l'algorithme de Ford-Fulkerson.



On utilisera d'abord un parcours de graphe en largeur (pour comparer avec un exercice du prochain TP) puis un parcours en profondeur.

Exercice 3. *Retrouver un résultat du cours*

Redémontrer le résultat préliminaire du théorème de Ford-Fulkerson. On considère un graphe valué (E, Γ, c) avec une seule source s et un seul puits t . Montrer que quelque soit le flot (f) réalisable et quelque soit la coupe (X, \bar{X}) , on a :

$$v(f) = \sum_{\substack{i \in X \\ j \in \bar{X}}} f_{ij} - \sum_{\substack{i \in X \\ k \in \bar{X}}} f_{ki}.$$

Déduire de ce résultat que la dernière équation (celle correspondant au puits) se déduit des $n - 1$ autres équations. Aide : utiliser $\bar{X} = \{t\}$.

Exercice 4. *Une méthode de débruitage d'images binaires³*

On considère une image binaire α constituée de n pixels noirs ou blancs, que l'on représentera de la façon suivante $\alpha : \{1, \dots, n\} \rightarrow \{\pm 1\}, i \mapsto \alpha_i$ (à chaque pixel i on associe sa couleur α_i (-1 pour du noir et 1 pour du blanc (ou l'inverse))). L'image originelle est supposée formée de zones blanches et noires bien délimitées mais elle est bruitée ce qui nous donne l'image α . On aimerait déterminer une image "restaurée" $\sigma : \{1, \dots, n\} \rightarrow \{\pm 1\}, i \mapsto \sigma_i$, en minimisant une énergie du type :

$$J(\sigma) := - \sum_{1 \leq i \leq n} \alpha_i \sigma_i - \sum_{1 \leq i, j \leq n} \gamma_{i,j} \sigma_i \sigma_j,$$

où les $\gamma_{i,j} \geq 0$ sont donnés (en pratique $\gamma_{i,j} > 0$ si les pixels i et j sont voisins et $\gamma_{i,j} = 0$ sinon).

Pour comprendre comment la minimisation de cette énergie peut fonctionner, il faut examiner l'influence des deux termes :

- le premier tend à rendre l'image σ proche de α (si le deuxième terme est absent il est clair qu'il faut $\sigma_i = \alpha_i$ pour minimiser J soit $\sigma = \alpha$);
- le deuxième terme est un terme de régularisation : sa minimisation tend à ce que des pixels voisins aient la même couleur⁴.

3. La méthode exposée dans cet exercice est due à *D. Greig, B. Porteous, et A. Seheult* dans un article paru en 1989 (Exact maximum a posteriori estimation for binary images. Journal of Royal Statistical Society.).

4. Fixez un pixel i qu'on supposera blanc ($\alpha_i = 1$) et considérez que ses voisins j sont noirs ($\alpha_j = -1$) le premier terme incite à avoir $\sigma_i = 1$ mais pour minimiser le second terme il faut choisir $\sigma_i = -1$.

Ainsi si une zone noire de l'image bruitée comporte peu de pixels blancs, ces derniers seront certainement changés en noir et inversement. Le but de l'exercice est de montrer que la minimisation de J correspond à trouver une coupe minimale d'un certain graphe.

1. Pourquoi peut-on choisir $\gamma_{i,j} = \gamma_{j,i}$ sans perte de généralité ?
2. À toute image σ , on associe la partition de $\{1, \dots, n\}$ formée des deux ensembles :

$$S^+ = \{i : \sigma_i = 1\}, S^- = \{i : \sigma_i = -1\}$$

(on notera (A^+, A^-) la partition correspondante pour l'image bruitée α). On construit le graphe orienté dont les sommets sont $\{1, \dots, n\}$ et les $n(n-1)$ arêtes⁵ orientées sont tous les arcs (i, j) , $1 \leq i, j \leq n$, $i \neq j$ munis chacun de leur capacité $\gamma_{i,j}$. Montrer que :

$$\sum_{1 \leq i, j \leq n} \gamma_{i,j}(1 - \sigma_i \sigma_j) = 4C(S^+, S^-)$$

où $C(S^+, S^-)$ désigne la capacité de la coupe (S^+, S^-) , i.e. $C(S^+, S^-) = \sum_{i \in S^+} \sum_{j \in S^-} \gamma_{i,j}$. On

adopte la convention $C(S^+, S^-) = 0$ au cas où l'un des deux sous-ensembles S^+ ou S^- serait vide (il n'y a pas encore de source, ni de puits dans ce graphe). *Aide* : partir de l'expression de gauche et scinder chaque somme en une somme sur S^+ et une autre sur S^- .

3. Dédurre de ce résultat l'expression suivante :

$$\frac{1}{4}J(\sigma) = -\frac{1}{4} \sum_{1 \leq i \leq n} \alpha_i \sigma_i + C(S^+, S^-) + Cte, \text{ avec } Cte = -\frac{1}{4} \sum_{1 \leq i, j \leq n} \gamma_{i,j}.$$

4. L'idée est de rajouter au graphe précédent, une source s et un puits t (on notera $\mathcal{S}^+ = \{s\} \cup S^+$ et $\mathcal{S}^- = \{t\} \cup S^-$) ainsi que certains arcs (s, i) et (i, t) , $i \in \{1, \dots, n\}$ adéquats munis de capacités bien choisies, de sorte que :

$$C(\mathcal{S}^+, \mathcal{S}^-) = -\frac{1}{4} \sum_{1 \leq i \leq n} \alpha_i \sigma_i + C(S^+, S^-) + Cte' \quad (1)$$

à une constante additive (Cte') près.

- (a) En supposant que l'on puisse effectivement créer ce dernier graphe, montrer que la minimisation de J (trouver σ^* vérifiant $J(\sigma^*) \leq J(\sigma), \forall \sigma$) revient à trouver la/une coupe minimale $C(\mathcal{S}_*^+, \mathcal{S}_*^-)$ de ce graphe et donc à colorier en blancs tous les pixels de \mathcal{S}_*^+ et en noir tous ceux de \mathcal{S}_*^- .
- (b) Montrer (en utilisant la définition d'une coupe) que :

$$C(\mathcal{S}^+, \mathcal{S}^-) = \sum_{i \in \mathcal{S}^-} c(s, i) + \sum_{i \in \mathcal{S}^+} c(i, t) + C(S^+, S^-) \quad (2)$$

où $c(i, j)$ désigne la capacité de l'arc (i, j) . Rmq : d'après l'équation (1) il faudra donc réussir à "identifier" (à une constante près) les deux premiers termes de droite avec $-\frac{1}{4} \sum_{1 \leq i \leq n} \alpha_i \sigma_i$.

- (c) *Construction du graphe final* :

- la source s est reliée à tous les noeuds i correspondant à des pixels blancs de l'image α , i.e. on rajoute les arcs $(s, i), \forall i \in A^+$, chacun muni d'une capacité de $1/2$;
- tous les noeuds i correspondant aux pixels noirs de α sont reliés au puits t , i.e. on rajoute les arcs $(i, t), \forall i \in A^-$, chacun muni aussi d'une capacité de $1/2$.

5. En fait la plupart de ces arêtes sont fictives car correspondent à $\gamma_{i,j} = 0$.

Montrer que cette construction répond bien à la question posée c'est à dire qu'on a bien :

$$\sum_{i \in S^-} c(s, i) + \sum_{i \in S^+} c(i, t) = -\frac{1}{4} \sum_{1 \leq i \leq n} \alpha_i \sigma_i + Cte', \text{ avec } Cte' = \frac{n}{4} \quad (3)$$

Pour cela, on pourra passer par les étapes suivantes :

i. En utilisant la définition du graphe final, montrer que :

$$\sum_{i \in S^-} c(s, i) + \sum_{i \in S^+} c(i, t) = \frac{1}{2} \sum_{i \in S^- \cap A^+} 1 + \frac{1}{2} \sum_{i \in S^+ \cap A^-} 1$$

et aussi que :

$$\sum_{i \in S^- \cap A^+} 1 = \sum_{i \in S^- \cap A^+} -\alpha_i \sigma_i, \text{ et } \sum_{i \in S^+ \cap A^-} 1 = \sum_{i \in S^+ \cap A^-} -\alpha_i \sigma_i$$

ii. À l'aide de l'identité :

$$n = \sum_{i \in S^- \cap A^+} 1 + \sum_{i \in S^+ \cap A^-} 1 + \sum_{i \in S^+ \cap A^+} 1 + \sum_{i \in S^- \cap A^-} 1$$

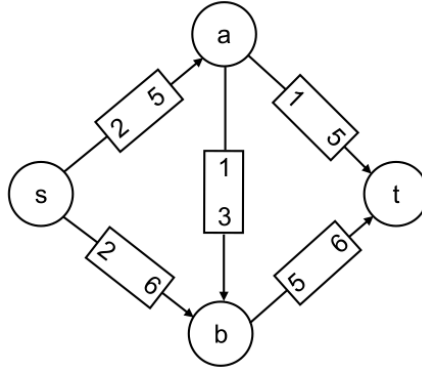
(et en remplaçant les 1 par $\pm \alpha_i \sigma_i$ en fct du domaine de sommation) montrer que :

$$\sum_{1 \leq i \leq n} \alpha_i \sigma_i = n + 2 \sum_{i \in S^- \cap A^+} \alpha_i \sigma_i + 2 \sum_{i \in S^+ \cap A^-} \alpha_i \sigma_i$$

et en déduire l'équation (3).

Exercice 5. Flot maximal avec capacités supérieures et inférieures

On considère le graphe suivant $G = (E, \Gamma, (\{\alpha_{ij}\}, \{\beta_{ij}\}))$ valué par des capacités supérieures et inférieures (non nécessairement nulles).



On rappelle qu'un flot $\{f_{ij}\}$ est réalisable sur G si $\alpha_{ij} \leq f_{ij} \leq \beta_{ij}$ pour tout $(i, j) \in \Gamma$.

1. *Flot réalisable sur G .*

- Considérer le changement de variable $f'_{ij} = f_{ij} - \alpha_{ij}$ pour se ramener au cas de flots positifs dans un graphe auxiliaire G' associé à G . Ce graphe G' possède une seule source s' et un seul puits t' de sorte que déterminer un flot réalisable sur G est équivalent à déterminer un flot maximal sur G' pour lequel toutes les arêtes sortant de la source s' et toutes les arêtes arrivant au puits t' sont saturées.
- Déterminer le flot maximal sur G' par l'algorithme de Ford-Fulkerson. Initialiser le flot $\{f'_{ij}\}$ en tenant compte des capacités des arêtes liées à la source s' et au puits t' de G' .
- En déduire un flot réalisable sur G .

2. Expliquer comment modifier l'algorithme de Ford-Fulkerson pour prendre en compte les capacités inférieures.
3. Déterminer le flot maximal sur G .

Exercice 6. *Réseau commercial*

Une entreprise gère un réseau commercial qui est composé de 3 guichets de prises de commandes (G_1, G_2, G_3), de 2 ateliers de préparation des commandes (A_1, A_2) et de deux centres de distribution (C_1, C_2). Le réseau est constitué de la façon suivante :

- chaque guichet peut passer commande aux 2 ateliers mais le faible nombre de personnels aux guichets limite à 20 le nombre maximum de commandes passées en un jour entre un guichet et un atelier.
- L'atelier A_1 fournit uniquement le centre C_1 et A_2 fournit uniquement C_2 .
- Le centre C_2 peut transférer sur C_1 une partie de la production reçue, sans modifier sa capacité de distribution.
- Les capacités journalières de prises de commandes des guichets G_1, G_2 et G_3 sont de 30, 30 et 10 respectivement.
- Les capacités journalières de préparation des commandes des ateliers A_1 et A_2 sont de 10 et 60 respectivement.
- Les capacités journalières de distribution des centres C_1 et C_2 sont de 30 et 50.

L'entreprise souhaite évaluer la capacité journalière du réseau c'est-à-dire le nombre maximum de commandes qui peuvent être prises, préparées et distribuées.

1. Etablir le graphe associé au réseau décrit ci-dessus.
2. Utiliser l'algorithme de Ford-Fulkerson pour déterminer le nombre maximum de commandes que l'entreprise peut traiter par jour. Interpréter les résultats obtenus.
3. Déterminer une coupe minimale.