TD #18 — Graphes #3

Exercice 1. *Chemin de plus forte probabilité*

On considère un graphe orienté pondéré G, dont les pondérations $\in [0,1]$ dénotent des probabilités ; la probabilité d'un chemin dans G est donnée par le **produit** des probabilités de ses arcs. Étant donnés deux sommets v, w de G, on souhaite trouver un chemin $v \rightsquigarrow w$ (que l'on supposera toujours exister) de probabilité maximale.

1. Proposez une *réduction* de ce problème à celui d'une recherche de plus court chemin pondéré dans le sens usuel. (On pourra pour cela supposer que l'on est capable de calculer exactement dans $\mathbb{R}$...)

Exercice 2. *FW : détection et reconstruction de cycles de poids négatif*

On considère un graphe orienté pondéré (sur les arcs) G (sans boucle).

1. Montrez l'invariant pour la boucle principale de FW que si G ne contient pas de cycle de poids strictement négatif **de sommets intermédiaires dans** $[\![k]\!]$, les entrées hors diagonale $\boldsymbol{D}_{i,j}$ de la matrice des distances calculée par l'algorithme contiennent les longueurs (pondérées) de (plus-court) chemins **élémentaires** $i \rightsquigarrow j$ de sommets intermédiaires dans $[\![k]\!]$.
2. Montrez que si G possède un cycle de poids strictement négatif, soit $i \rightsquigarrow k \rightsquigarrow i$ l'un parmi ces cycles minimisant k l'on aura que $\boldsymbol{D}_{i,k} + \boldsymbol{D}_{k,i} < 0$ à la $k^{\text{ième}}$ itération de la boucle principale.
3. Conclure sur la capacité de FW à détecter la présence d'un cycle de poids strictement négatif, et à reconstruire un éventuel tel cycle.

Exercice 3. *Rencontre au milieu*

Soit $G = S, A$ un graphe (éventuellement orienté) pondéré à pondération positive et $v, w \in S$, on cherche un sommet m se trouvant le plus « au milieu » de v et w dans G, c'est à dire un qui minimise le maximum des distances (pondérées) $d(v,m)$ et $d(w,m)$.

1. Proposez une modification de l'algorithme « de Dijkstra » qui permette de résoudre ce problème efficacement. En particulier, pour G représenté par tableau de listes d'adjacence, le coût de l'algorithme modifié doit être un $O(\#A \log \#S + \#S)$.
2. Implémentez cet algorithme en OCaml (vous pouvez supposer qu'une structure de données de file de priorité minimum (par exemple impérative) est déjà disponible).

Exercice 4. *Graphe équivalent minimum d'un DAG*

Soit $G = S, A$ un graphe orienté acyclique connexe, on définit sont *graphe équivalent (pour l'accessibilité) minimum* comme un sous-graphe $G' = S, A' \subseteq A$ de G de même fermeture transitive que G (c'est à dire tel que pour tout $v, w \in S$, il existe un chemin $v \rightsquigarrow w$ dans G' ss.'il en existe un dans G) et minimisant $\#A'$. Autrement dit, G' est de même fermeture transitive que G, et ce n'est le cas d'aucun de ses sous-graphes stricts $G'' = S, A'' \subset A'$.

1. Donnez des graphes équivalents minimum de G_1 et G_2 respectivement définis par :
   - $G_1 : S = [\![3]\!], A = \{(0,1),(0,2),(2,1)\}$
   - $G_2 : S = [\![5]\!], A = \{(0,1),(0,3),(1,2),(3,4),(4,2))\}$

Dans toute la suite de l'exercice, on suppose (sans perte de généralité) que $S = [\![n]\!]$ pour un certain n.

On définit une fonction r sur les graphes de la façon suivante : si $G = S, A$ possède un triplet $u, v, w \in S$ tels que (u, w), $(v, w) \in A$ et qu'il existe un chemin $u \rightsquigarrow v$ dans G, alors on définit $r(G)$ comme le graphe $G' = S, A \setminus \{(u, w)\}$ obtenu en supprimant celui de ces (éventuellement multiples) triplets minimum pour l'ordre lexicographique. Sinon l'on définit $r(G)$ comme G lui-même.

On propose alors l'algorithme « glouton » suivant pour calculer G' : on renvoie le premier point-fixe obtenu en itérant r à partir de G.

2. Montrez que cet algorithme termine.
3. Montrez que cet algorithme renvoie bien un graphe équivalent minimum.
4. Montrez que le graphe renvoyé par l'algorithme précédent est unique. (Plus généralement, montrez que G possède un unique graphe équivalent minimum.)
   Indication : on peut procéder par l'absurde, de façon assez similaire à la question précédente :(
5. Quelle conséquence cette unicité peut-elle avoir sur une implémentation de l'algorithme glouton ?

On suppose maintenant que G est représenté par une matrice d'adjacence $\boldsymbol{A}$ à valeur dans $\{\top, \bot\}$ et que l'on dispose également de la matrice d'adjacence $\boldsymbol{A}^F$ de sa fermeture transitive (également à valeur dans $\{\top, \bot\}$).

6. Soit $\odot$ la multiplication de matrices dans le semi-anneau $\{\vee, \wedge\}$, (on remplace l'addition (resp. la multiplication) par le « OU » (resp. le « ET ») logique) et $\boldsymbol{A}^{2+} := \boldsymbol{A} \odot \boldsymbol{A}^F$, montrez que $\boldsymbol{A}^{2+}_{v,w} = \top$ (vaut « VRAI ») ss.'il existe un chemin $v \rightsquigarrow w$ dans G de longueur **au moins deux**.
7. Déduisez de cela (et de tout ce qui précède) un algorithme qui calcule le graphe équivalent minimum d'un graphe orienté acyclique G de n sommets en temps $O(n^3)$.