TD numéro 8

Flots dans un réseau

Préliminaires :

Soit $G = (V, E)$ un graphe orienté avec deux sommets distingués, une *source* s et un *puits* t, et une capacité positive $cap(u, v)$ pour chaque arête (u, v) de E. S'il n'y a pas d'arêtes entre deux sommets, on dira que la capacité est nulle. Un *flot* de G est une fonction à valeur réelle sur l'ensemble des paires de sommets telle que :

Anti-symétrie : $f(u, v) = -f(v, u)$. Si $f(u, v) > 0$, il y a un flot de u vers v.

Contraintes de capacité : $f(u, v) \leq cap(u, v)$. S'il existe une arête (u, v) telle que $f(u, v) = cap(u, v)$, on dit que le flot *sature* (u, v).

Conservation du flot : pour tout sommet v différent de s et t, $\sum_{u \in V} f(u, v) = 0$.

La valeur d'un flot f, noté $|f|$ est le flot sortant de la source, $\sum_{u \in V} f(s, u)$. Le *problème du flot maximal* consiste à trouver un flot de valeur maximale, appelé un *flot maximal.*

Une (s, t)-coupe d'un graphe est une partition de l'ensemble des sommets V en deux ensembles disjoints X et $\bar{X} = V \setminus X$ tels que $s \in X$ et $t \in \bar{X}$. La capacité d'une coupe $cap(X, \bar{X}) = \sum_{v \in X, w \in \bar{X}} cap(v, w)$. Une (s, t)-coupe de capacité minimale est appelée une *coupe minimal.* Si f est un flot, et $X, \bar{X}$ une coupe, le *flot à travers la coupe* est $f(X, \bar{X}) = \sum_{v \in X, w \in \bar{X}} f(v, w)$

Montrer que pour tout flot f, le flot à travers toute coupe $X, \bar{X}$ est égal à la valeur du flot et en déduire que le flot maximal est inférieur à la coupe minimale.

Les *capacités résiduelles* pour un flot f est une fonction sur les paires de sommets donnée par $res(u, v) = cap(u, v) - f(u, v)$. On peut pousser $res(u, v)$ unités de flots supplémentaires de u à v en augmentant $f(u, v)$ et en diminuant $f(v, u)$. Quand on change $f(u, v)$, on doit aussi changer $f(v, u)$ pour maintenir la propriété d'antisymétrie. Le *graphe résiduel* R_f pour un flot f est le graphe $G_f = (V, E_f)$ et une arête de capacité $res(u, v)$ pour toutes paires telles que $res(u, v) > 0$. Un *chemin améliorant* pour f est un chemin de s à t dans R. La *capacité résiduelle* de p, noté $res(p)$ est la valeur minimale de $res(u, v)$ pour (u, v) une arête de p.

Montrer que les conditions suivantes sont équivalentes (Théorème max-flot min-cut) :

1. f est un flot maximal
2. il n'y a pas de chemin améliorant pour f
3. $|f| = cap(X, \bar{X})$ pour une coupe $X, \bar{X}$

Montrer alors que Ford-Fulkerson s'exécute en au plus $O(m|f^*|)$, où m est le nombre d'arêtes du graphe d'origine et f^*, le flot maximal.

Montrer que si les capacités sont des entiers, il existe un flot maximal entier.

Montrer qu'en commençant par le flot zéro, il est possible de construire un flot maximal en au plus $m = |E|$ étapes, chacune d'elles augmente le flot sur un seul chemin dans le graphe d'origine.

Montrer qu'en augmentant toujours le chemin de capacité résiduelle maximale, les valeurs successives du flot convergent vers le flot maximal. Si les capacités sont des entiers, la méthode trouve un flot maximal en $O(m \log c)$ chemins améliorants où c est le maximum des capacités des arêtes.

Une arête d'un chemin améliorant est *critique* si sa capacité résiduelle réalise le minimum de $cap(u, v) - f(u, v)$ quand (u, v) décrit les arêtes du chemin améliorant. On notera $d(s, u)$ la fonction distance du sommet s au sommet u dans le graphe résiduel, chaque arête comptant pour une unité.

1) Rappeler le principe de Edmonds-Karp.

2) Soit $d_i(s, v)$ la distance de la source s au sommet v, comptée en nombre d'arêtes dans le graphe résiduel R_i calculé après la i-ième itération de l'algorithme de Edmonds-Karp. Montrer que, pour v fixé, $d_i(s, v)$ est une fonction croissante de i.

3) Un couple de sommets distincts (u, v) constitue une arête i-critique si (u, v) est une arête du graphe résiduel R_i mais pas une arête du graphe résiduel R_{i+1}. Montrer que si (u, v) est i-critique et j-critique, $i < j$ alors $d_i(s, u) < d_j(s, u)$.

4) En déduire une estimation de la complexité de Edmonds-Karp.

Exercice 1 : Plan d'évacuation

On considère un graphe non orienté $G = (V, E)$ ayant n sommets, et deux sous-ensembles Y et Z (non nécessairement disjoints) ; les éléments de Y sont les places et les éléments de Z les issues.

Un *plan d'évacuation* est un ensemble de chemins dont les sommets de tous les chemins sont disjoints dans le graphe, menant d'une place à une issue. Déterminer un plan d'évacuation *optimal* consistant à trouver un plan d'évacuation contenant le nombre maximal de chemins.

Un *problème de flot avec contraintes sur les sommets* est un problème de flot classique auquel on ajoute des contraintes supplémentaires : à chaque sommet i est associé un nombre s_i exprimant que la somme des flots entrants en i doit être inférieure ou égale à s_i.

1. Réduire le problème du plan d'évacuation à un problème de flot avec contraintes sur les sommets.
2. Réduire le problème de flot avec contraintes sur les sommets à un problème de flot.
3. En déduire un algorithme de recherche d'un plan d'évacuation optimal.

Exercice 2 : Recouvrement d'un graphe par des chemins

Un recouvrement d'un graphe orienté est un ensemble P de chemins à sommets disjoints tel que tout sommet soit inclus dans exactement un chemin de P. Les chemins peuvent commencer et se terminer n'importe où, et éventuellement ne contenir qu'un sommet. Un recouvrement *minimal* est un recouvrement par chemins contenant le moins de chemins possibles.

1. Donner un algorithme efficace dans le cas où d'un graphe orienté sans circuit $G = (S, A)$. On pourra considérer le graphe $G' = (S', A')$ en supposant que $S = \{1, 2, \ldots, n\}$ et

$$\begin{aligned} S' &= \{x_0, x_1, \ldots, x_n\} \cup \{y_0, y_1, \ldots, y_n\} \\ A' &= \{(x_0, x_i) : i \in S\} \cup \{(y_i, y_0) : i \in S\} \cup \{(x_i, y_j) : (i, j) \in A\} \end{aligned}$$

   En déduire la valeur du recourement minimal en fonction du nombre de sommets du recouvrement et du nombre de chemins.
2. Pourquoi l'algorithme ne fonctionne pas pour les graphes orientés contenant des circuits ?

Exercice 3 : Résistance aux pannes

Un paramètre représentant la résistance aux pannes d'un graphe non orienté G est le nombre minimum d'arêtes à retirer de G pour le déconnecter. Calculer ce nombre en utilisant des algorithmes de flot.

Exercice 4 : Couplage parfait - Théorème de Hall

Un *couplage parfait* est un couplage dans lequel chaque sommet est couvert. Soit $G = (S, A)$ un graphe biparti non orienté ayant la partition de sommets $S = L \cup R$, où $|L| = |R|$. Pour un $X \subseteq S$ quelconque, on définit le *voisinage* de X par

$$V(X) = \{y \in S : (x, y) \in A \text{ pour un certain } x \in X\}$$

c'est-à-dire l'ensemble des sommets adjacents à un certain membre de X. Démontrer le théorème de Hall : il existe un couplage parfait dans G si et seulement si $|E| \leq |V(E)|$ pour tout sous-ensemble $E \subseteq L$.

Exercice 5 : Couplage d'un graphe biparti d-régulier

Un graphe biparti est d-régulier si chaque sommet a exactement d voisins. Soient L et R les sommets des deux parties de G. Montrer que si G est d-régulier, alors $|L| = |R|$. Montrer que tout graphe biparti d-régulier a un couplage de taille $|L|$ en argumentant que la coupe minimale du réseau de flot associé a une capacité de $|L|$.

Exercice 6 : Algorithme randomisé pour calculer une coupe minimale

Soit G un multi-graphe (éventuellement plusieurs arêtes entre chaque paire de sommets) de coût unitaire. On répète l'étape suivante : choisir une arête uniformément au hasard et fusionner les deux sommets. S'il y a plusieurs arêtes entre les paires de sommets nouvellement formés, il faut toutes les garder. Les arêtes entre les sommets qui ont fusionné sont éliminées. Ce processus de fusion est appelé une *contraction*. Après une contraction, le nombre de sommets diminue de 1 et une contraction ne réduit pas la taille de la coupe minimale de G car toute coupe du graphe après la i-ième contraction est une coupe du graphe d'origine. L'algorithme continue tant qu'il ne reste pas 2 sommets. A la fin, on compte le nombre d'arêtes entre les deux sommets restants.

Soit $\mathcal{E}_i$ l'événement "de ne pas choisir une arête de la coupe à la i-ième itération" pour $1 \leq i \leq n-2$. Montrer que $\Pr[\cap_{i=1}^{n-2} \mathcal{E}_i] \geq \frac{2}{n(n-1)}$.

Montrer qu'en réitérant l'algorithme un certain nombre de fois, la probabilité de ne pas trouver une coupe minimale est $< 1/e$.

Calculer efficacement une contraction d'un graphe.