TD Algo2 – session 7 – Algorithmes élémentaires pour les graphes

21 mars 2025

Objectifs d'apprentissage :
— analyser l'impact des représentations des graphes sur des opérations algorithmiques simples ;
— manipuler les cas particuliers et les propriétés des algorithmes de parcours ;
— concevoir des algorithmes s'appuyant sur ces représentations et parcours ;

Exercice 1 : Degré

Étant donnée une représentation par listes d'adjacences d'un graphe orienté connexe (une seule composante connexe), quelle la complexité en temps pour calculer le degré sortant de tous les sommets ? Même question avec la représentation par matrice d'adjacences. Et pour calculer les degrés entrants ?

Exercice 2 : Carré d'un graphe

Le carré d'un graphe orienté $G = (V, E)$ est le graphe $G^2 = (V, E^2)$ tel que $(u, w) \in V^2$ si et seulement s'il existe un $v \in V$, tel que $(u, v) \in V$ et $(v, w) \in V$. Autrement dit, G^2 possède un arc entre u et w chaque fois que G contient un chemin de longueur deux exactement entre u et w. Décrire des algorithmes efficaces permettant de calculer G^2 à partir de G, quand G est représenté par listes d'adjacences, puis par matrice d'adjacences. Analyser le temps d'exécution de vos algorithmes.

Exercice 3 : Arbre de parcours en largeur

On rappelle qu'un arbre de parcours en largeur d'un graphe orienté connexe $G = (V, E)$ est l'arbre obtenu par un parcours en largeur sur un sommet origine $s \in V$: c'est un sous-ensemble des arcs $E_\pi \subseteq E$ tels que, pour chaque sommet $v \in V$, le chemin unique dans E_π de s à v soit un plus court chemin dans G.

Donner un exemple de graphe ayant un arbre de parcours qui ne puisse être obtenu en exécutant un parcours en largeur sur G, quel que soit l'ordre des sommets dans chaque liste d'adjacences.

Exercice 4 : Labyrinthe

Soit $G = (V, E)$ un graphe non orienté et connexe. Donner un algorithme en $O(E)$ pour calculer un chemin dans G qui traverse chaque arête dans E exactement une fois dans chaque direction. Décrire comment on peut trouver son chemin dans un labyrinthe si on dispose d'une grande quantité de pièces de monnaie.

Exercice 5 : Trou noir

Quand on utilise la représentation par matrice d'adjacences, la plupart des algorithmes de graphes s'exécutent en $\Theta(V^2)$, mais il y a des exceptions. Montrer qu'il est possible de déterminer en $O(V)$ si un graphe orienté contient un trou noir, c'est-à-dire un sommet de degré entrant $V - 1$ et de degré sortant 0 si on utilise une représentation par matrice d'adjacences.

Exercice 6 : Table de hachage

Supposons qu'au lieu d'une liste chaînée, chaque entrée du tableau $G.adj[u]$ soit une table de hachage contenant les sommets v pour lesquels $(u, v) \in E$, les collisions étant résolues par chaînage. Dans l'hypothèse d'un hachage indépendant uniforme, si toutes les recherches d'arêtes ont la même probabilité, quel est le temps attendu pour déterminer si une arête se trouve dans le graphe ? Quels sont les inconvénients de ce système ? Proposer une autre structure de données pour chaque liste d'arêtes qui résout ces problèmes. Votre solution présente-t-elle des inconvénients par rapport à la table de hachage ?

Exercice 7 : Arbre de parcours en profondeur

Le docteur Alex Ample souhaite trouver des graphes qui montrent que les conjectures suivantes sont fausses. La première conjecture affirme que s'il existe un chemin entre u et v dans un graphe orienté

G, et si $u.d < v.d$ lors d'un parcours en profondeur de G, alors v est un descendant de u dans la forêt de parcours en profondeur obtenue. La seconde conjecture indique que s'il existe un chemin entre u et v dans un graphe orienté G, alors tout parcours en profondeur donne forcément $v.d \leq u.f$. Alex Ample se rend compte qu'un même graphe avec 3 sommets et avec le même parcours peut servir dans les deux cas. Lequel ?

Exercice 8 : Catcheurs

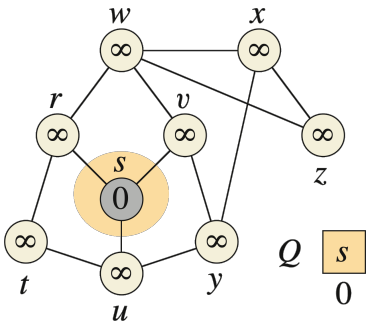
Il existe deux types de catcheurs : les "bons" et les "méchants". Entre deux catcheurs donnés, il peut ou non y avoir une rivalité. Supposer que l'on ait n catcheurs et une liste de r paires de catcheurs telle qu'il y ait rivalité entre les deux membres de chaque paire. Donner un algorithme à temps $O(n + r)$ qui détermine s'il est possible de désigner certains catcheurs comme étant les bons et les autres comme étant les méchants, de telle sorte que chaque rivalité oppose un bon à un méchant. S'il est possible de faire ce genre de classification, votre algorithme doit la calculer.

Exercice 9 : Arbre binaire

Donner une représentation par listes d'adjacences pour un arbre binaire complet à 7 sommets. Donner la représentation équivalente par matrice d'adjacences. On suppose que les sommets sont numérotés de 1 à 7 comme dans un tas binaire.

Exercice 10 : Parcours en largeur

Donner les valeurs d et π qui résultent d'un parcours en largeur du graphe non orienté de la figure suivante, en prenant pour origine le sommet u.



Exercice 11 : Diamètre

Le diamètre d'un arbre $T = (V, E)$ est donné par $\max_{u,v \in V} d(u, v)$. Autrement dit, le diamètre est la plus grande de toutes les distances de plus court chemin dans l'arbre. Donner un algorithme efficace permettant de calculer le diamètre d'un arbre, et analyser son temps d'exécution.

Exercice 12 : Indépendance

Montrer que, dans un parcours en largeur, la valeur $u.d$ affectée à un sommet u est indépendante de l'ordre dans lequel les sommets apparaissent dans chaque liste d'adjacences.