TD d'Algorithmique et structures de données
Graphes

Équipe pédagogique Algo SD

1 Connexité

On cherche dans un premier exercice à déterminer si un graphe orienté est connexe.

1.1 Arbres

On suppose dans un premier temps disposer d'un graphe restreint à un arbre orienté père-fils.

1. Proposez une fonction récursive `exploration(G, N)` renvoyant le nombre de nœuds atteignables à partir d'un nœud N donné dans un graphe G. Votre fonction devra être indépendante de la structure choisie pour représenter le graphe.

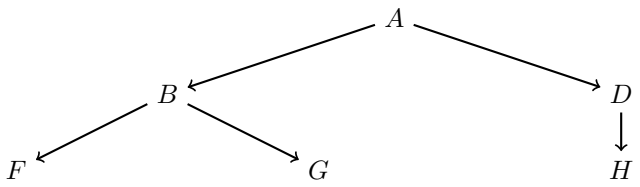

FIGURE 1 – arbre exemple

2. Dessiner le graphe d'appel pour le graphe de la figure 1 et un appel sur le nœud A.
3. Annoter chaque appel du graphe d'appel par son coût propre (sans compter les coûts induits par la récursion) dans le cas où l'on utilise des listes de voisins.
4. Quel est le coût au pire cas dans le cas où le graphe est stocké comme listes de voisins ?
5. Quel est le coût au pire cas dans le cas où le graphe est stocké comme matrice d'adjacence ?

1.2 Graphe généraux

On considère à présent des graphes quelconques.

1. En modifiant votre fonction d'exploration proposez une fonction `tous_atteignables(G, D)` renvoyant *Vrai* si tous les nœuds de G sont atteignables à partir de D et *Faux* sinon.
2. Quel est le coût au pire pour les différentes structures de graphes ?
3. Dans quel cas préférer les listes de voisins à une matrice d'adjacence ?

2 DAG

On s'intéresse à compter le nombre de chemins entre deux sommets s et t d'un graphe orienté sans cycle (DAG pour *Directed Acyclic Graph*).

1. En remarquant que le nombre de chemins de s à t est égal à la somme pour chaque successeur s' de s du nombre de chemins entre s' et t, écrire une méthode `nbChemins(G, s, t)` qui répond au problème donné.
2. Quelle est la complexité de votre algorithme ?

3 Graphes colorés

On considère maintenant des graphes *non-orientés* dont les nœuds peuvent être colorés.

Nous ne considérerons ici qu'une seule couleur (rouge). Tout nœud est donc soit coloré (en rouge) soit sans aucune couleur.

La figure 2 donne un petit exemple d'un tel graphe. L'information relative à la couleur (coloré ou non) est stockée dans un dictionnaire C (une table de hachage par exemple) indicé par les identifiants des nœuds.

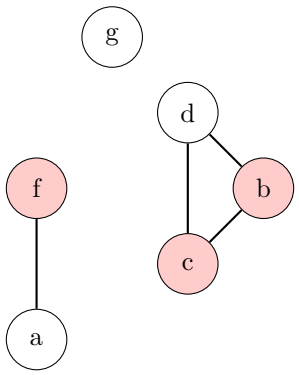

FIGURE 2 – Graphe coloré

On cherche à écrire une fonction `proba(G, C)` renvoyant la probabilité que 2 nœuds rouges de G tirés uniformément parmi les nœuds rouges soient connectés.

Sur notre exemple, les nœuds rouges sont b, c, f. On a donc une chance sur neuf de tirer aléatoirement b et b, qui sont connectés ; une chance sur neuf de tirer b et c qui sont également connectés ; une chance sur neuf de tirer b et f qui ne sont pas connectés... Au final, la probabilité de tirer deux nœuds rouges connectés est ici de $\frac{5}{9}$.

1. Proposez différents algorithmes de coûts plus ou moins élevés.