Algorithmique et Programmation

TD n° 5 : Parcours de graphes

École normale supérieure – Département d'informatique
algoL3@di.ens.fr

2016-2017

Exercice 1. Arbre des blocs

Les parcours en largeur et profondeur permettent de trouver les composantes connexes d'un graphe. Nous pouvons définir des notions de connexité plus élevées et dans certain cas, être en mesure de calculer les « composantes » pour ces notions. Dans cet exercice, nous verrons un exemple avec la 2-connexité.

Un *sommet séparant* d'un graphe est un sommet qui augmente le nombre de composantes connexes du graphe lorsqu'on l'enlève. (Pour un graphe G connexe, $G - v$ n'est pas connexe.) Un (sous)graphe est *2-connexe* s'il n'a pas de sommet séparant. Un *bloc* d'un graphe G est un ensemble maximal de sommets de G qui induisent un sous graphe 2-connexe.

1. Montrer que la racine r d'un arbre de parcours en profondeur est un sommet séparant si et seulement si r a au moins deux fils.
2. Montrer qu'un sommet non-racine v d'un arbre de parcours en profondeur est séparant si et seulement si v a un fils u tel qu'il n'y a pas d'arête (hors-arbre) d'un descendant de u vers un ancêtre propre de v.
3. Donner un algorithme linéaire qui trouve pour tout sommet v dans un arbre de parcours en profondeur, le plus haut (plus près de la racine) sommet que l'on peux atteindre en une seule arète hors-arbre à partir du sous-arbre de v (arbre de v et tous ses descendants).
4. Donner un algorithme pour trouver tous les sommets séparants d'un graphe connexe en temps linéaire.
5. Donner un algorithme pour trouver tous les blocs d'un graphe connexe en temps linéaire (à partir des sommets séparants et de l'arbre de parcours en profondeur du graphe).

Exercice 2. Circuit eulérien

Un *circuit eulérien* d'un graphe orienté fortement connexe G à n sommets et m arêtes est un cycle qui traverse chaque arête exactement une fois. Un tel parcours existe toujours si le degré entrant de chaque sommet de V est égal à son degré sortant. Donner un algorithme de complexité $O(n+m)$ pour trouver un parcours eulérien dans un tel graphe.

Exercice 3. Coloration de graphes

Étant donné un graphe non-orienté $G = (V, E)$, une k-coloration de G est une fonction $c : V \to \{1, \ldots, k\}$ associant à chaque sommet u de G une couleur $c(u)$, telle que $c(u) \neq c(v)$ pour toute arête uv de G. Un graphe pour lequel il existe une k-coloration est dit *k-colorable* (ou *colorable avec k couleurs*).

1. Montrer que chaque graphe connexe et acyclique est 2-colorable.

2. Donner un algorithme polynomial qui détermine si un graphe est 2-colorable et, si oui, donne une 2-coloration de celui-ci. Quelle est la complexité de l'algorithme (on représentera un graphe par liste d'adjacences) ?

3. Montrer en construisant un algorithme, que les graphes de degré maximum Δ sont $(\Delta + 1)$-colorables. Quelle est la complexité de l'algorithme ?

4. Montrer en construisant un algorithme polynomial que tout graphe 3-colorable à n sommets peut être colorié avec $O(\sqrt{n})$ couleurs.

   On peut utiliser l'algorithme de la question 2 en remarquant que dans un graphe 3-colorable, pour tout sommet u donné, le sous-graphe correspondant aux sommets v tel que uv est une arête est lui 2-colorable.