Algorithmique et programmation — TD des 14 et 17 décembre 1999

Graphes

1. Composantes biconnexes

   Dans un graphe connexe non-orienté, on dit qu'un cycle est simple s'il ne passe pas deux fois par un même sommet. On définit une relation $\sim$ sur l'ensemble des arêtes, où deux arêtes sont en relation si elles peuvent être impliquées dans un même cycle simple.

   (a) Montrer que c'est une relation d'équivalence. On appelle *composantes biconnexes* les classes d'équivalence.

   (b) Proposer un algorithme pour calculer les composantes biconnexes à l'aide d'un parcours en profondeur.

2. Circuits eulériens

   Un chemin d'un graphe orienté (V, E) est une suite d'arcs se succédant. Un circuit est un chemin fermé. Un circuit eulérien est un circuit qui parcourt chaque arc exactement une fois.

   (a) À quelle condition un graphe admet-il un circuit eulérien ?

   (b) Proposer un algorithme de complexité $\mathcal{O}(E)$ pour trouver un tel circuit s'il existe.

3. Arborescences

   Une arborescence de racine r est un graphe orienté (fini) dont r est un sommet et tel que pour tout autre sommet x il existe un unique chemin de r à x. On représente souvent une arborescence en numérotant ses sommets et en indiquant pour chacun quel est son père.

   (a) Quelle est la différence entre un arbre et une arborescence ?

   (b) Si les sommets sont numérotés de 1 à n, on dit qu'une arborescence est préfixe lorsque l'ensemble des descendants de tout sommet i est un intervalle dont le minimum est i. Trouver un algorithme de renumérotation qui rend une arborescence préfixe.

4. Arborescence recouvrante

   Étant donné un graphe orienté G, une arborescence recouvrante est un graphe partiel de G qui est une arborescence et qui passe par tous les sommets. On dit qu'un graphe est enraciné en r s'il existe une arborescence recouvrante de racine r.

   (a) Trouver un algorithme qui dit si un graphe est enraciné.

5. Chemin de poids minimal

   On suppose que chaque arc est valué par un réel. Le poids d'un chemin est la somme des poids de ses arcs.

   (a) Proposer un algorithme pour calculer un chemin de poids minimal entre deux sommets, dans le cas où les poids des arcs ne sont jamais négatifs.

   (b) Proposer un algorithme pour trouver s'il existe un cycle de poids négatif.

   (c) Proposer un algorithme pour calculer un chemin de poids minimal dans le cas général.

6. Arborescences recouvrantes de poids minimal

   On suppose que chaque arc est valué par un réel strictement positif. Le poids d'une arborescence est la somme des poids de ses arcs.

   (a) Proposer un algorithme pour trouver une arborescence recouvrante de poids minimal pour un graphe non-orienté.

   (b) Regarder le cas des graphes orientés.

7. Arborescence des dominants

   On suppose que le graphe (orienté) G est acyclique et enraciné. On dit qu'un sommet en domine un autre s'il est dans tout chemin de la racine à cet autre sommet.

   (a) Montrer que la relation de domination décrit une arborescence.

   (b) Trouver un algorithme qui calcule cette arborescence.

8. Réduction transitive

   Rappel : la clôture transitive d'un graphe (orienté) est un graphe ayant les mêmes sommets et dont deux sommets sont reliés par une arête s'il y a un chemin les reliant dans le graphe de départ.
   Une réduction transitive est un graphe avec un nombre minimal d'arêtes et même clôture transitive.

   (a) Trouver une condition pour que la réduction transitive soit unique.

   (b) Trouver un algorithme de calcul d'une réduction transitive.

9. Graphes bipartites

   Un graphe non-orienté est dit bipartite si V se découpe en deux parties telles que toute arête relie un sommet dans chaque partie. On appelle *couplage* une partie $M \subset E$ telle que chaque sommet du graphe appartient à au plus une arête de M. Un couplage est dit maximal si son cardinal l'est. Un couplage est parfait si tous les sommets en font partie. Pour toute partie $X \subset V$, on note $V(X)$ l'ensemble des sommets adjacents à au moins un sommet de X.

   (a) Proposer un algorithme pour décider si un graphe est bipartite.

   (b) Proposer un algorithme pour trouver un couplage maximal.

   (c) Montrer le théorème de Hall : un graphe admet un couplage parfait si, et seulement si $\forall X \subset V, \quad |X| \geqslant |V(X)|$.