Master-2 d'Informatique 2014–2015
Complexité algorithmique appliquée

Feuille 2 : Autour du problème Vertex Cover

Une couverture par sommets (Vertex Cover, sigle VC) d'un graphe $G = (V, E)$ est un sous-ensemble $C \subseteq V$ des sommets du G tel que pour toute arête $(u, v) \in E$ au moins une de ses extrémités u, v appartient à C. Le problème VC est le suivant :

- Donnée : Un graphe non-orienté $G = (V, E)$ et un entier p.
- Question : Existe-t-il une couverture de G de taille p ?
- Version : Trouver la plus petite couverture du graphe G.

Il est connu que le problème VC est NP-complet.

Exercice 1 Montrer que $C \subseteq V$ est une couverture de G si et seulement si $D = V \setminus C$ est un *ensemble indépendant* dans G : aucune paire de sommets $u, v \in D$ n'est lié par une arête.

Exercice 2 Soit $H = (V, F)$ le *graphe complémentaire* du $G = (V, E)$: il possède le même ensemble de sommets V, et une paire de sommets (u, v) est une arête de H si et seulement si elle n'est pas une arête de G. Montrer que $C \subseteq V$ est une couverture du graphe G si et seulement si $D = V \setminus C$ est une clique dans le graphe H.

Exercice 3 Montrer la propriété suivante : soit F une forêt, et (u, v) une arête telle que le degré de v est 1. Il existe alors une couverture de F de taille minimale qui contient le sommet u, mais pas v.

Exercice 4 Soit T un arbre. Proposer un algorithme récursif qui calcule une couverture de taille minimale de T. Le temps de calcul de votre algorithme doit être *linéaire*.

Exercice 5 On propose l'algorithme suivant pour la couverture des graphes connexes quelconques G :

a) Calculer un arbre T correspondant à un parcours en profondeur de G.

b) Retourner l'ensemble S des sommets non-feuilles de T. (On voit l'arbre T comme arbre orienté, et les sommets non-feuilles sont les sommets qui ont au moins un successeur dans l'arbre.)

1. Justifier que cet l'algorithme retourne une couverture.
2. Trouver un graphe sur lequel l'algorithme ci-dessus retourne une solution dont la taille est deux fois la taille optimale d'un VC.
3. Justifier que l'algorithme ci-dessus est 2-approché.

Exercice 6 Un *couplage maximal* dans un graphe $G = (V, E)$ est un ensemble d'arêtes $M \subseteq E$ qui vérifie les deux propriétés suivantes : (a) deux arêtes différentes de M n'ont pas d'extrémité en commun ; (b) on ne peut pas ajouter une autre arête à M sans violer la propriété précédente.

1. Montrer sur un exemple qu'un couplage maximal peut ne pas être un couplage le plus grand possible.
2. Montrer que l'ensemble des extrémités d'un couplage maximal M est une couverture du graphe G.
3. Soit M un couplage maximal d'un graphe G. Montrer que toute couverture de G contient au moins autant de sommets qu'il y a d'arêtes dans M.

Remarque : Un théorème de König affirme que pour les graphes bipartis le nombre d'arêtes dans un plus grand couplage est égal au nombre de sommets dans la plus petite couverture. Ceci mène à un algorithme polynomial pour VC pour les graphes bipartis.

Exercice 7 On propose l'algorithme suivant pour la couverture des graphes connexes quelconques G :

a) Prendre une arête arbitraire (u, v) du graphe G et mettre dans la couverture ses *deux* extrémités u et v.

b) Enlever du graphe ces deux sommets ainsi que toutes les arêtes incidentes, et continuer.

Montrer que cet algorithme est 2-approché.

Exercice 8 Montrer les propriétés suivantes :

1. Soit G un graphe à n sommets, et d le degré maximum de ses sommets. Si G possède une couverture de taille k, alors il a au plus kd arêtes.
2. Soit G un graphe et $e = (u, v)$ une arête quelconque. Alors G possède une couverture de taille k si et seulement si soit $G \setminus \{u\}$, soit $G \setminus \{v\}$ possède une couverture de taille $k - 1$.
   Remarque : $G \setminus \{u\}$ est le graphe G sans le sommet u et ses arêtes incidentes.

Proposer un algorithme récursif qui donne une solution exacte au problème VC, basé sur les deux propriétés précédentes. Votre algorithme devra avoir un temps de calcul de $O(2^k \cdot kn)$. Justifier le temps de calcul en établissant une formule récursive sur le temps $t(n, k)$ en fonction de n et k.

Exercice 9 On propose l'algorithme suivant pour la couverture des arbres :

1. On colorie les sommets en noir et blanc ; toute arête possède une extrémité noire et une extrémité blanche.
2. On considère l'ensemble des sommets noirs et l'ensemble des sommets blancs, et on choisit comme couverture celui qui est plus petit.

Montrer sur un exemple que cet algorithme est très loin d'être optimal.

Exercice 10 On propose deux versions d'algorithme GLOUTON :

1. Choisir un sommet de degré maximum ; inclure ce sommet dans la couverture ; éliminer toutes les arêtes incidentes au sommet choisi ; continuer de la même manière avec le sous-graphe restant.
2. Choisir dans la couverture les sommets en ordre décroissant de leurs degrés jusqu'à ce que toutes les arêtes soient couvertes. (On ne diminue pas les degrés des sommets au cours de l'application de l'algorithme.)

Question facile : Montrer sur un exemple qu'aucun des deux algorithmes n'est optimal.
Question difficile : Montrer sur un exemple que le coefficient d'approximation pour chacun des deux algorithmes peut être arbitrairement grand.