Université Paris Diderot
L3 Informatique

Algorithmique
Année 2008-2009, 1^{er} semestre

TD n°8

Arbres couvrants de poids minimum.

Dans ce TD, $G = (V, E)$ est un graphe fini, non-orienté, sans boucle, dont les arêtes sont pondérées. La fonction $\omega : E \rightarrow \mathbb{R}^+$ est la fonction de poids des arêtes. Le poids de G, noté $\omega(G)$, est la somme des poids des arêtes de G (c'est-à-dire $\omega(G) = \sum_{(x,y)\in E(G)} \omega(x, y)$).

La Minute Historique

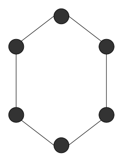
Le problème de trouver un arbre couvrant de poids minimum est un problème très ancien. Et aussi bizarre que cela puisse paraître, c'est un problème qui est même plus vieux que l'informatique elle même. En effet en 1926, Otakar Borůvka fut un des premiers (si ce n'est le premier) à fournir un algorithme pour trouver "efficacement" un arbre couvrant de poids minimum. La motivation d'alors était de concevoir un réseau de distribution électrique pour la Moravie du Sud. Avec la contrainte évidente : que cela coûte le moins possible. Ce n'est qu'au milieu des années 50 et l'avènement de l'informatique que ce problème est revenu à la mode. C'est à Joseph B. Kruskal (1956) et à Robert Prim (1957) que nous devons les algorithmes que nous allons étudier aujourd'hui.

Définitions

Arbre recouvrant de poids minimum (ACPM) : un arbre T est un arbre couvrant de G si $T = (V, A)$ est un arbre dont l'ensemble des sommets V est celui de G et l'ensemble des arêtes A est un sous ensemble des arêtes de G ($A \subseteq E$). De plus, un arbre couvrant T est dit de poids minimum si pour tout arbre couvrant T_i de G, on a $\omega(T) \leqslant \omega(T_i)$.

Cocycle : soit $S \subset V$. Le cocycle de S dans G, noté $\delta(S)$, est l'ensemble des arêtes qui ont exactement une extrémité dans S et l'autre dans $V \setminus S$ (c'est-à-dire $\delta(S) = \{(x, y)|(x, y) \in E$ avec $x \in S$ et $y \in V \setminus S\}$).

Exercice 1 [ACPM] Trouvez sur les graphes suivants un arbre couvrant de poids minimum.



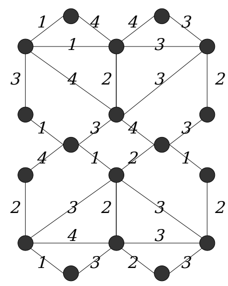


FIGURE 1 – (a) Graphe valué unitairement, (b) graphe valué

Pour chaque graphe, l'ACPM que vous avez trouvé est-il unique ?

Exercice 2 [Arête minimum] Soient $G = (V, E)$ un graphe valué et $T = (V, A)$ un ACPM. Montrez que l'arête de poids minimum de G appartient à T. Qu'en est-il de la réciproque ?

Exercice 3 [Algorithme de Prim : borne inférieure]

L'algorithme de Prim consiste à choisir un sommet x, le placer dans l'ensemble S, puis choisir l'arête $e = (y, z)$ (avec $y \in S$) du cocycle de S qui a un poids minimum ; placer z dans S et répéter cette opération jusqu'à ce que $S = V$.

Données : $G = (V, E)$ un graphe, ω une fonction de pondération des arêtes.
Résultat : $T = (V, A)$ un ACPM de G.

$S \leftarrow v \in V$
tant que $S \neq V$ **faire**
  $e \leftarrow min(\delta(S))$ où $e = (x, y)$ avec $x \in S$
  $A \leftarrow A \cup \{e\}$
  $S \leftarrow S \cup \{y\}$
retourner $T = (V, A)$

Algorithme 1 : Algorithme de Prim

1. Procédez à une exécution de l'algorithme de Prim sur l'exemple de l'exercice 1.
2. Quelle structure de donnée doit-on employer pour implémenter cet algorithme ?
3. Quelle est la complexité de l'algorithme ?
4. Montrez que l'algorithme de Prim, dans un modèle comparatif, est équivalent au tri.
5. Que peut-on conclure de la borne inférieure de complexité pour l'algorithme de Prim ?

Exercice 4 [Arbre couvrant de poids max.] On se propose maintenant d'étudier le problème de trouver l'arbre couvrant T de poids maximum : T est toujours couvrant mais cette fois pour tout arbre couvrant T' de G, $\omega(T) \geqslant \omega(T')$.

Est-ce que ce problème est relié à l'ACPM ?

Donnez un algorithme pour trouver un tel arbre. Quelle est sa complexité ?

Exercice 5 [Arbre couvrant de poids min, $\times$.] On se propose maintenant d'étudier le problème de trouver l'arbre couvrant de poids minimum quand la fonction de valuation est définie par :

$$\omega(G) = \prod_{(x,y) \in E(G)} \omega(x, y)$$

Est-ce que ce problème est relié à l'ACPM ?

Donnez un algorithme pour trouver un tel arbre. Quelle est sa complexité ?

Exercice 6 [Algorithme de Kruskal]

Données : $G = (V, E)$ un graphe, ω une fonction de pondération des arêtes.
Résultat : $T = (V, A)$ un ACPM de G.

1 $S \leftarrow \emptyset$
2 $A \leftarrow \emptyset$
3 $ET \leftarrow$ L'ensemble des arêtes de E triées de manière croissante
4 **tant que** $S \neq V$ **faire**
5   $e = (x, y) \leftarrow$ **RetirerEnTete**(ET)
6   **si** *e ne crée pas de cycle dans* $T = (S, A)$ **alors**
7     $S \leftarrow S \cup \{x\} \cup \{y\}$
8     $A \leftarrow A \cup \{e\}$
9 **retourner** $T = (S, A)$

Algorithme 2 : Algorithme de Kruskal

Procédez à une exécution de l'algorithme de Kruskal sur l'exemple de l'exercice 1.

En supposant que le test de la ligne 6 se fait en $f(n, m)$, où f est une fonction inconnue, calculer la complexité de l'algorithme en fonction de n, m et f.