Arbres couvrants, cycles eulériens

RO en L3 Miage

Arbres

Définition 1. Un *arbre* est un graphe *connexe* sans cycle.

Théorème 1. *Les énoncés suivants sont équivalents.*

1. *G est un arbre ;*
2. *Pour tout couple de sommets $x, y \in V(G)$, il existe une chaîne élémentaire unique entre x et y ;* « chaînes uniques »
3. *G est connexe, et si on enleve n'importe quelle arête, le graphe devient non connexe ;* « minimalement connexe »
4. *G est acyclique, et si on rajoute une nouvelle arête à G, le nouveau graphe contiendra un cycle ;* « maximalement acyclique »
5. *G est connexe et $|V(G)| = |E(G)| + 1$.* « la formule d'Euler »

Définition 2. Soit G un graphe. Un *arbre couvrant de G* est un sous-graphe couvrant de G qui est un arbre.

Proposition 2. *Tout graphe connexe contient un arbre couvrant.*

Démonstration. Soit G un graphe connexe. Retirons de G, tant qu'il est possible, une arête qui ne coupe pas le graphe (le graphe reste connexe). On obtient un sous-graphe partiel T qui est connexe par la condition sur les arêtes, et il n'a pas de cycles puisque s'il y aurait un cycle, on pourrait enlever une arête du cycle sans couper le graphe. T est donc un arbre. □

Arbres couvrants de poids minimum

Définition 3. Soit G un graphe connexe avec une fonction de poids $w : E(G) \to \mathbb{R}$. Un *arbre couvrant de poids minimum* de G est un arbre couvrant $T \subseteq G$ qui minimise $\sum_{e \in E(T)} w(e)$.

Exemple : réalisation d'un réseau électrique ou informatique entre différents points, deux points quelconques doivent toujours être reliés entre eux (connexité) et on doit minimiser le coût de la réalisation.

Le problème de trouver un arbre couvrant de poids minimum peut être résolu avec l'algorithme de Kruskal.

Justification de l'algorithme. L'algorithme s'arrête au pire lorsque toutes les arêtes du graphe ont été considérées. Comme le graphe est fini, l'algorithme s'arrête au bout d'un nombre fini d'opérations.

Montrons qu'à la fin de l'exécution de l'algorithme, le graphe partiel $T = G[A]$ engendré par l'ensemble A d'arêtes est un arbre couvrant minimum. D'abord T est un arbre couvrant de G, par construction

Algorithme 1 : Kruskal

Entrées : Un graphe connexe $G = (V, E)$ et une fonction de poids w sur E.
Sorties : Un arbre couvrant T de G de poids minimum.
F, A : sous-ensembles de E
$F := E$;
$A := \emptyset$;
tant que $|A| < |V| - 1$ **faire**
  trouver $e \in F$ tel que $w(e)$ soit minimum;
  $F := F - \{e\}$;
  si $G[A \cup \{e\}]$ *est acyclique* **alors**
    $A := A \cup \{e\}$;
retourner $T := G[A]$;

puisqu'il est acyclique et vérifie la relation $m = n - 1$ qui vient de la condition de sortie de la boucle « tant que » ($|A| = |V| - 1$).

Il reste à montrer que T est de poids minimum. Soit T_0 un arbre couvrant de G de poids minimum, et soit e_1 l'arête la plus légère dans $E(T_0) - E(T)$. Le graphe $T \cup \{e_1\}$ contient un cycle C (voir le théorème du début du cours). Comme T_0 est acyclique, il existe au moins une arête $e_2 \in E(C) - E(T_0)$. Si $w(e_1) < w(e_2)$, l'algorithme de Kruskal aurait choisi l'arête e_1 au lieu de e_2. Donc, $w(e_1) \geq w(e_2)$.

Soit $T_1 = (T_0 - \{e_1\}) \cup \{e_2\}$; comme $w(T_1) \leq w(T_0)$, T_1 est un (autre) arbre couvrant de G de poids minimum, tel que $|E(T) \cap E(T_1)| = |E(T) \cap E(T_0)| + 1$.

Répéter cet argument pour l'arbre T_1, T_2, etc. À chaque étape, T_i est un arbre couvrant de G de poids minimum, tel que $|E(T) \cap E(T_{i+1})| = |E(T) \cap E(T_i)| + 1$. Finalement $T_k = T$ (pour $k \leq n - 1$), donc T est un arbre couvrant de G de poids minimum. □

Complexité. La complexité de l'algorithme, dominée par l'étape de tri des arêtes, est $O(m \log n)$ avec m le nombre d'arêtes et n le nombre de sommets du graphe G.

Cycles eulériens et le problème du postier chinois

Dans cette section on permet des arêtes parallèles !

Graphes eulériens

La ville de Königsberg (aujourd'hui Kaliningrad) est construite autour de deux îles situées sur le Pregel et reliées entre elles par un pont. Six autres ponts relient les rives de la rivière à l'une ou l'autre des deux îles. Le problème consiste à déterminer s'il existe ou non une promenade dans les rues de Königsberg permettant, à partir d'un point de départ au choix, de passer une et une seule fois par chaque pont, et de revenir à son point de départ, étant entendu qu'on ne peut traverser le Pregel qu'en passant sur les ponts.

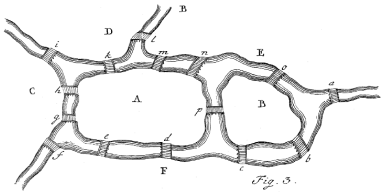

FIGURE 1: Les ponts de Königsberg.

Le problème a été résolu en 1736 par Leonhard Euler. Il a représenté chaque masse terrestre par un sommet d'un graphe, et chaque pont par une arête. Notons que le graphe correspondant au problème des sept ponts de Königsberg contient trois sommets de degré 3 et un sommet de degré 5. Euler a montré le théorème suivant.

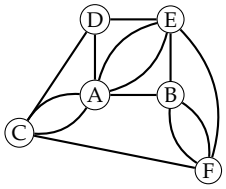

FIGURE 2: Représentation graphique des ponts de Königsberg.

Soit G un graphe. Un cycle $C \subseteq G$ est *eulérien* si C passe par chaque arête de G une et une seule fois. On appelle un graphe avec un cycle eulérien un *graphe eulérien*.

Théorème 3. *Un graphe G est eulérien si et seulement si G est connexe, et tout sommet de G est de degré pair.*

Démonstration. Pour la nécessité, soit G un graphe eulérien avec un cycle eulérien C et soit v un sommet de G quelconque. Chaque fois que C passe par v, il faut utiliser deux arêtes incidentes à v. Donc, le degré de v est pair.

La suffisance est une conséquence de l'algorithme de Fleury. □

Une *chaîne eulérienne* est une chaîne $P \subseteq G$ telle que P passe par chaque arête de G une et une seule fois.

Théorème 4. *Un graphe G contient une chaîne eulérienne si et seulement si G est connexe et précisément deux sommets de G sont de degré impair.*

Démonstration. Soit x et y les sommets de G de degré impair. Le graphe $G + xy$ est eulérien ; soit C un cycle eulérien de $G + xy$. Alors $C - xy$ est une chaîne eulérienne. □

Algorithme 2 : Fleury

Entrées : Un graphe eulérien $G = (V, E)$ et un sommet u de G.
Sorties : Un cycle eulérien C de G qui commence et termine à u.

```
// Variables
```
F : sous-ensemble de E
C : chaîne dans G
x : sommet de G
```
// Initialisation
```
$C := u$;
$x := u$;
$F := G$;
tant que $\partial_F(x) \neq 0$ **faire**
    choisir une arête $e = xy \in \partial_{G[F]}(x)$, où e n'est pas un isthme de $G[F]$ sauf si c'est la seule possibilité;
    $C := Cey$;
    $x := y$;
    $F := F - \{xy\}$;
retourner C;