Recherche Opérationnelle 1A
Théorie des graphes
Arbres + arbre couvrant de coût minimum

Zoltán Szigeti

Laboratoire G-SCOP
INP Grenoble, France

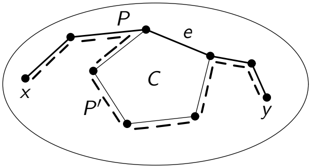
Connexité

Lemme 1

Si e est une arête d'un cycle C d'un graphe connexe G, alors $G - e$ est connexe.

Démonstration

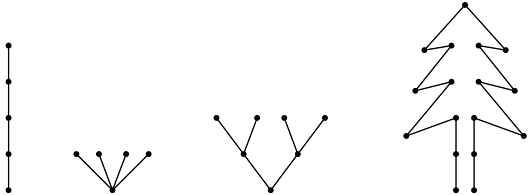
1. $\forall x, y \in V(G)$, il existe une (x, y)-chaîne P car G est connexe.
2. Si $e \notin E(P)$, alors P est une (x, y)-chaîne dans $G - e$.
3. Si $e \in E(P)$ alors en remplaçant dans P l'arête e par la chaîne $C - e$ on obtient une (x, y)-chaîne P' dans $G - e$.
4. $G - e$ est donc connexe.



Arbres, forêts

Définition

- arbre : graphe connexe, sans cycle.
- forêt : graphe sans cycle.



Remarques

- Chaque composante connexe d'une forêt est un arbre.
- Tout graphe partiel d'un arbre est une forêt.

Arbres : propriétés utiles

Lemme 2

Soit G un arbre à $n \geq 2$ sommets.

(a) G contient un sommet de degré 1.

(b) En supprimant un sommet v de degré 1 de G on obtient un arbre.

Démonstration

(a)
1. Puisque G est connexe et $n \geq 2$, par l'EXO 2.3, $\forall u \in V : d(u) \geq 1.$
2. Puisque G est sans cycle, par l'EXO 2.6(b), $\exists v \in V : d(v) \leq 1.$
3. Par conséquent, $d(v) = 1.$

(b)
1. $G - v$ est connexe : $\forall u, w \in V(G - v)$,
   1. puisque G est connexe, $\exists (u, w)$-chaîne élémentaire P dans G,
   2. par $v \neq u, w$, $d(x) \geq 2$ $\forall x \in V(P) \setminus \{u, w\}$ et $d(v) = 1$, on a $v \notin V(P)$,
   3. d'où P est une (u, w)-chaîne dans $G - v$.
2. $G - v$ est sans cycle : puisque G est sans cycle.
3. $G - v$ est donc un arbre.

Arbres : nombre d'arêtes

Théorème 1

Le nombre m d'arêtes d'un arbre à n sommets est égal à $n-1$.

Démonstration

1. Par récurrence sur n.
2. Si $n = 1$, alors $m = 0 = 1 - 1 = n - 1$.
3. On suppose que c'est vrai pour tous les arbres à $n - 1 \geq 1$ sommets.
4. Soit G un arbre à n sommets.
5. Par Lemme 2, $\exists v \in V(G) : d(v) = 1$, et $G' = G - v$ est un arbre
6. tel que $m' = m - 1$ et $n' = n - 1$.
7. En vertu de l'hypothèse de récurrence: $m = m' + 1 = n' = n - 1$.

Arbre couvrant

Définition

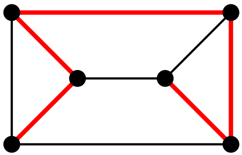
arbre couvrant de G : graphe partiel de G qui est un arbre.

Théorème 2

Un graphe G contient un arbre couvrant $\Longleftrightarrow$ G est connexe.

Démonstration de nécessité :

Si H est un arbre couvrant de G, alors H est connexe, donc G l'est aussi.



Arbre couvrant

Démonstration de suffisance :

1. Supposons que G est un graphe connexe.
2. Soit H un graphe partiel connexe de G avec un nombre minimum d'arêtes.
3. Si H contenait un cycle C, alors pour une arête e de C, par Lemme 1, $H - e$ serait un graphe partiel connexe de G contenant moins d'arêtes que H ce qui contredirait la minimalité de H.
4. Par conséquent, H est un graphe partiel de G connexe et sans cycle, donc un arbre couvrant de G.

Arbre couvrant de coût minimum

Motivation

Etant donnés un réseau et un coût pour chaque connexion directe, trouver un sous-réseau fonctionnel de coût minimum.

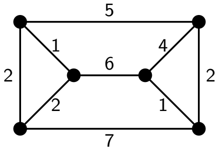
Définition

Etant donnés $G = (V, E)$, $c : E \to \mathbb{R}$, $F \subseteq E$, c-coût de F : $\sum_{e \in F} c(e)$.

Problème

Etant donnés un graphe connexe et un coût (≥ 0) pour chaque arête

1. trouver un graphe partiel connexe du graphe de coût minimum $\Longleftrightarrow$
2. trouver un arbre couvrant du graphe de coût minimum.



Arbre couvrant de coût maximum

Remarque

H est un arbre couvrant de G

1. de c-coût maximum $\Longleftrightarrow$
2. de $(-c)$-coût minimum $\Longleftrightarrow$
3. de $(C-c)$-coût minimum où $C = \max\{c(e) : e \in E(G)\}$, puisque, par le Théorème 1, chaque arbre couvrant contient le même nombre d'arêtes.

Trouver un arbre couvrant de coût minimum

Algorithme de Kruskal

ENTRÉE: Un graphe connexe G, une fonction de coût c sur les arêtes de G.
SORTIE : Un arbre couvrant H de G de coût minimum.

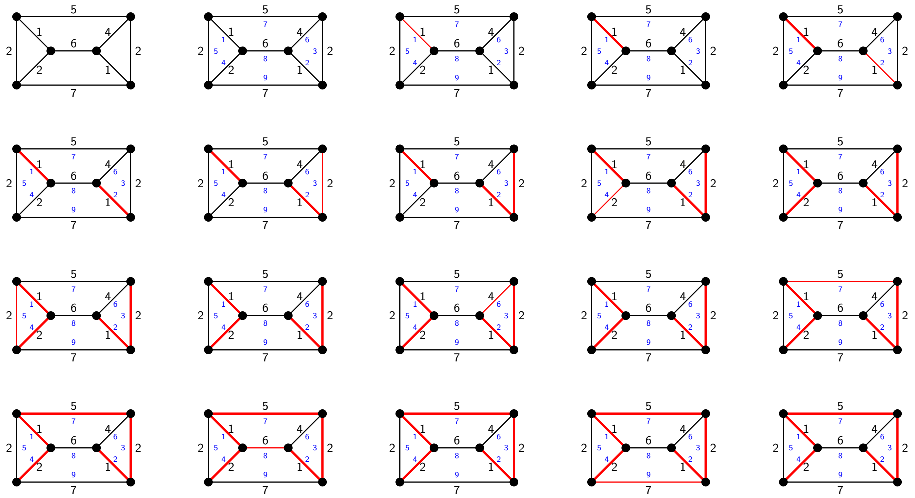
Etape 0: *Prétraitement des données.*
Trier les arêtes de G par ordre de coût non-décroissant :
$c(e_1) \leq c(e_2) \leq \ldots \leq c(e_m)$.

Etape 1: *Initialisation.*
$H_0 := (V, F_0)$ où $F_0 := \emptyset$.

Etape 2: *Construction de l'arbre.*
Pour $i = 1$ à m faire :
si $H_{i-1} + e_i$ est une forêt, alors $F_i := F_{i-1} + e_i$,
sinon $F_i := F_{i-1}$,
$H_i := (V, F_i)$.

Etape 3: *Fin de l'algorithme.*
$H := H_m, F := F_m$, STOP.

Trouver un arbre couvrant de coût minimum : Exemple



H est un arbre couvrant de coût $= 11$.

Justification 1

On commence par montrer que H est un arbre couvrant de G.

1. H est une forêt :
   1. H_0 est une forêt, donc chaque H_i, en particulier $H(= H_m)$, l'est aussi.
2. H est connexe : soit $X \subset V(G), X \neq \emptyset$.
   1. Puisque G est connexe, d'après le Théorème sur connexité, $\delta_G(X) \neq \emptyset$,
   2. soit donc $e_i = uv$ l'arête d'indice minimum dans cette coupe.
   3. Par le Théorème sur les chaînes, il n'y a pas de (u, v)-chaîne dans H_{i-1},
   4. $H_{i-1} + e_i$ est donc une forêt, et ainsi $e_i \in F_i \subseteq F_m$, d'où $\delta_H(X) \neq \emptyset$.
   5. D'après le Théorème sur connexité, H est connexe.

Algorithme de Kruskal

Justification 2

1. Soit $H' = (V, F')$ un arbre couvrant de coût minimum tq $|F \cap F'|$ soit maximum. Par Théorème 1, $|F| = n - 1 = |F'|$.
2. Si $F' \neq F$ soit i le plus petit indice tel que $e_i \in F \setminus F'$. $(F_{i-1} \subseteq F'.)$
3. $H' + e_i$ contient un cycle C car H' est connexe.
4. C contient $f' \in F' \setminus F$ car H est sans cycle.
5. $H'' = (V, F'') = H' + e_i - f'$ est connexe par Lemme 1, $|F''| = n - 1$ et ainsi par les Théorèmes 1 et 2, H'' est un arbre couvrant de G.
6. Par $F_{i-1} \cup f' \subseteq F'$ et principe de l'algorithme, $c(e_i) \leq c(f')$.
7. Par l'hypothèse et (6), $c(F') \leq c(F'') = c(F') + c(e_i) - c(f') \leq c(F')$.
8. F'' est aussi de coût minimum et $|F \cap F''| > |F \cap F'|$, contradiction.
9. Ainsi $F' = F$, et H est donc un arbre couvrant de coût minimum.

Algorithmes

Remarque

1. Principe de l'algorithme de Kruskal :
   ajouter la meilleure arête possible le graphe restant une forêt.
2. Principe de l'algorithme de Prim :
   ajouter la meilleure arête possible le graphe restant un arbre.
3. Principe de l'algorithme "pessimiste" :
   enlever la pire arête possible le graphe restant connexe.