

MTH6412B

Implémentation d'algorithmes de recherche opérationnelle

Arbres de recouvrement minimaux

Dominique Orban
dominique.orban@polymtl.ca

Mathématiques et Génie Industriel
École Polytechnique de Montréal

Automne 2020

Définitions

Rappel :

- Un *arbre* est un graphe non-orienté *acyclique* et *connexe*.

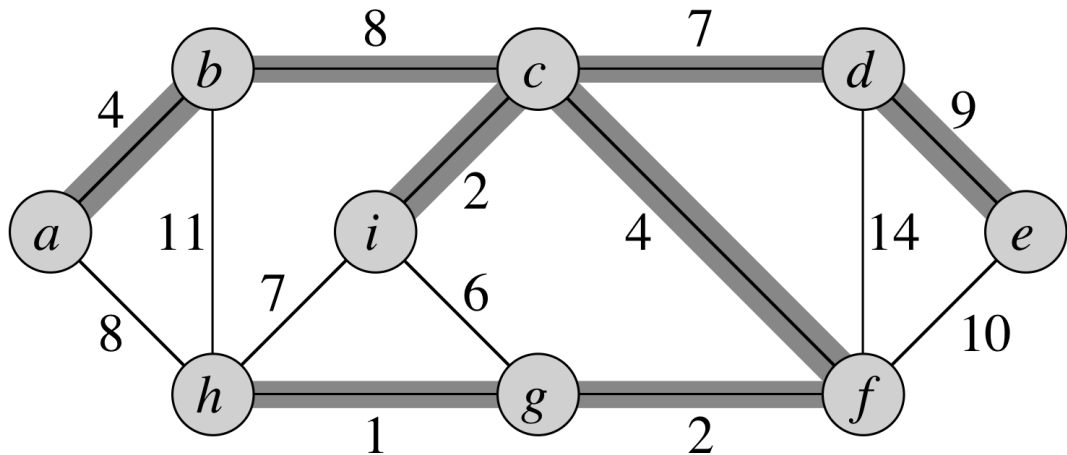
Définition (Arbre de recouvrement)

Un sous-graphe de $G = (S, A)$ qui possède le même ensemble de sommets S et qui constitue un arbre est un *arbre de recouvrement*. Parmi tous les arbres de recouvrement de G , un de ceux qui est de poids minimal est un *arbre de recouvrement minimal*.

Théorème

Si G est connexe, il admet un arbre de recouvrement. Plus généralement, tout graphe admet une forêt de recouvrement.

Arbre de recouvrement minimal : exemple



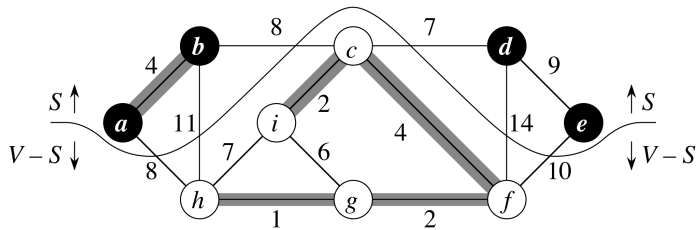
Arbre de poids 37. On en obtient un autre en remplaçant l'arête $[b,c]$ par $[a,h]$.

Quelques définitions

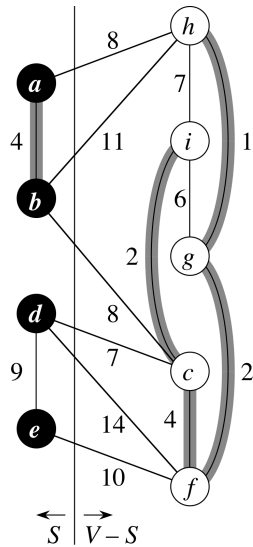
Définition

- ▶ Une coupe $(S', S \setminus S')$ du graphe non orienté $G = (S, A)$ est une partition de S ;
- ▶ l'arête $[s_i, s_j]$ traverse la coupe si $s_i \in S'$ et $s_j \in S \setminus S'$;
- ▶ une coupe respecte un sous-ensemble $A' \subseteq A$ d'arêtes si aucune arête de A' ne traverse la coupe ;
- ▶ une arête traversant une coupe est légère si elle est de poids minimal parmi toutes les arêtes traversant cette coupe.

Exemple



(a)



(b)

Comment faire pousser un arbre : méthode générique

Idée : à chaque étape, on a un ensemble d'arêtes qui forment un sous-ensemble d'un arbre de recouvrement minimal. Chaque itération ajoute une arête de façon à satisfaire cette propriété.

Théorème

Supposons que $G = (S, A)$ est connexe. Soit $A' \subseteq A$ un sous-ensemble d'arêtes qui fait partie d'un arbre de recouvrement minimal et $C = (S', S \setminus S')$ une coupe qui respecte A' . Soit $[s_i, s_j]$ une arête légère traversant la coupe C . Alors $A' \cup [s_i, s_j]$ fait encore partie d'un arbre de recouvrement minimal.

Exercice

L'arête de poids minimal dans G fait nécessairement partie d'un arbre de recouvrement minimal.

Idées principales de la construction

Notons A_k le sous-ensemble d'arêtes identifié après k itérations et $G_k = (S, A_k)$ le sous-graphe de G déterminé par A_k .

- ▶ G_k n'a jamais de cycle ;
- ▶ G_k est une forêt et chacune de ses composantes connexes est un arbre (certains peuvent ne contenir qu'un seul sommet — c'est le cas quand $k = 0$) ;
- ▶ chaque arête ajoutée connecte deux composantes connexes de G_k — G_{k+1} a donc une composante connexe de moins que G_k ;
- ▶ la méthode s'arrête quand G_k est connexe.

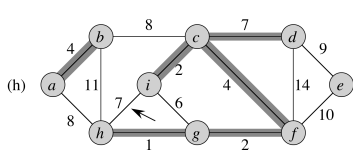
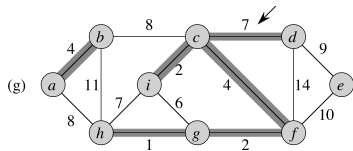
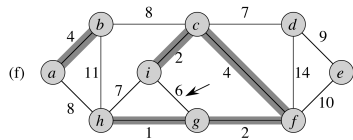
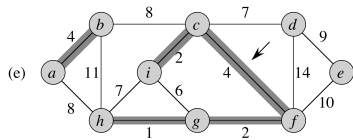
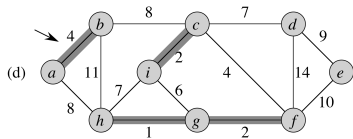
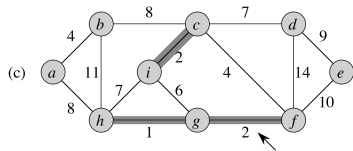
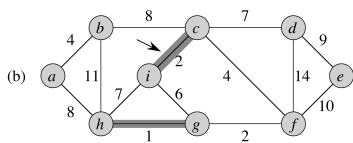
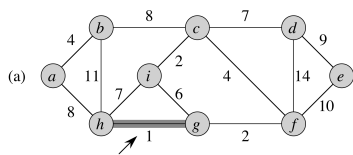
Résultat de base

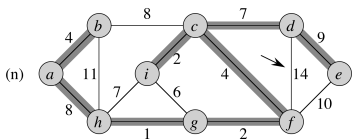
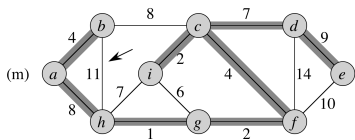
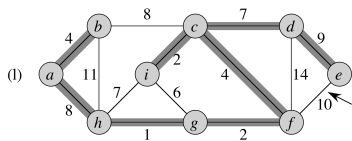
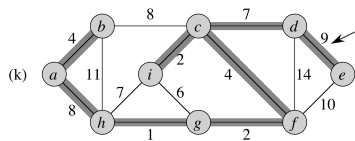
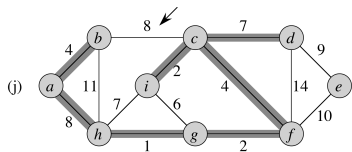
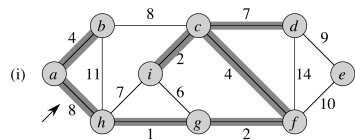
Corollaire

Supposons que $G = (S, A)$ est connexe. Soit $A' \subseteq A$ un sous-ensemble d'arêtes qui fait partie d'un arbre de recouvrement minimal et C une composante connexe de $G' := (S, A')$ (C est un arbre dans la forêt G'). Si $[s_i, s_j]$ est une arête légère connectant C à une autre composante de G' , alors $A' \cup [s_i, s_j]$ fait encore partie d'un arbre de recouvrement minimal.

L'algorithme de Kruskal

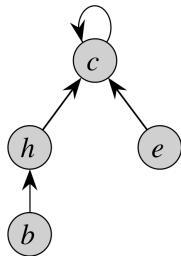
- ▶ On débute avec $A_0 = \emptyset$, $k = 0$. Chaque $s \in S$ forme une composante connexe.
- ▶ Trier les arêtes de A en ordre croissant de poids ;
- ▶ pour chaque arête $[s_i, s_j] \in A$, si s_i et s_j font partie de composantes connexes distinctes,
 - ▶ $A_{k+1} = A_k \cup [s_i, s_j]$;
 - ▶ réunir les composantes connexes contenant s_i et s_j ;
 - ▶ $k \leftarrow k + 1$.



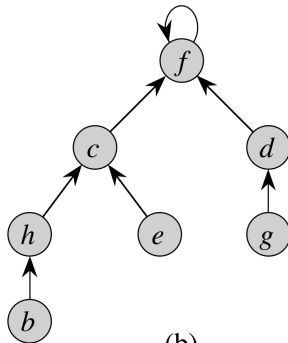


Forêt d'ensembles disjoints

- Pour représenter les composantes connexes d'un graphe ;
- chaque nœud « pointe » vers son parent ;
- la racine est son propre parent ;
- deux nœuds se trouvent dans le même ensemble s'ils ont la même racine ;
- l'union se fait en réaffectant le parent d'une racine.



(a)



(b)

Première partie de ce laboratoire

1. Implémenter une structure de données pour les composantes connexes ;
2. implémenter l'algorithme de Kruskal et le tester sur l'exemple des notes de laboratoire ;
3. tester votre implémentation sur diverses instances de TSP symétrique.