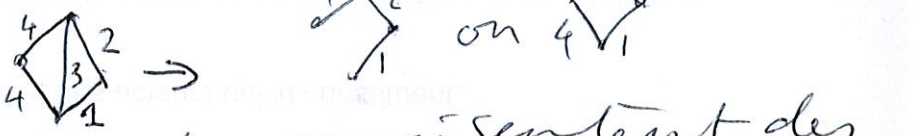
Arbre couvrant minimum

Anglais : minimum spanning tree

On suppose un graphe simple non orienté $G = (S, A)$ et on a une fonction de poids définie sur les arêtes $P : A \to \mathbb{R}$.

On suppose G connexe

Un arbre couvrant minimum (i.e., arbre couvrant de poids minimum) est un arbre couvrant (S, T) de G (i.e., $T \subseteq A$ et T forme un arbre) tel que $\sum_{a \in T} P(a)$ soit le plus petit possible.



P.ex. les sommets représentent des sites, et des arêtes représentent des circuits de communication, chacun ayant un coût. Pour interconnecter tous les sites avec un coût minimum, on restreint les arêtes à un arbre de poids minimum.

ND Si on complémente les poids $P(a) \mapsto$ (le $P(a)$) alors le problème devient de trouver un arbre couvrant de poids maximal.

$$\sum_{a\in T} (cte - P(a)) = cte \times \operatorname{card} T - \sum_{a\in T} P(a)$$

$$= cte\,(\operatorname{card} S - 1) - \sum_{a\in T} P(a)$$

Un algorithme d'A.C.M. a été donné par Borůvka en 1926. Son but était d'optimiser l'électrification de la Moravie du sud.

Les algorithmes de Kruskal et de Prim sont basés sur l'ajout d'arêtes dans l'ordre croissant de poids.

Soit $G=(S,A)$ un graphe non-orienté. Nous allons croître un sous-ensemble $R \subseteq A$ tel que (S,R) est un graphe partiel d'un arbre couvrant minimum : $\exists\, T \subseteq A, R \subseteq T$ et (S,T) est un A.C.M.

Clairement $\emptyset \subseteq A$ et $(S,\emptyset)$ est un graphe partiel de tout A.C.M.

Proposition. Soit $\{S_1, S_2\}$ une partition de S, et soit $R \subseteq A$ tel ~~qu'il existe $T \subseteq A$ avec $R \subseteq T$ et (S,T)~~ que $\forall a \in R$, les extrémités de a sont

soit toutes deux dans S_1, soit toutes deux dans S_2, et (S,R) est un graphe partiel d'un A.C.M.

Parmi toutes les arêtes ayant une extrémité dans S_1 et l'autre dans S_2, prenons b de poids minimum.

Alors $(S, R \cup \{b\})$ est un graphe partiel d'un arbre couvrant minimum.

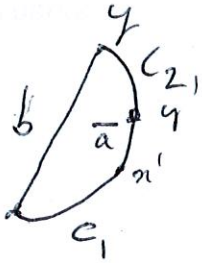
Preuve. On a $R \subseteq T \subseteq A$, où (S,T) est un ACM.

Si $b \in T$, alors $R \cup \{b\} \subseteq T$, OK.

Sinon, $\delta(b) = \{x, y\}$ avec $x \in S_1$ et $y \in S_2$, et comme $b \notin T$, il existe dans T une chaîne C d'extrémités x et y, dont les arêtes sont $\neq b$. Comme $x \in S_1$ et $y \in S_2$, cette chaîne C comprend une arête $\bar{a}$ ayant une extrémité x' dans S_1 et l'autre y' dans S_2.

Soit $T' = (T \setminus \{\bar{a}\}) \cup \{b\}$.

(S,T') est un arbre. $C = C_1 \bar{a} C_2$



et dont chaîne de T comprenant $\bar{a}$ peut être modifiée en remplaçant $\bar{a}$ par $c_1^{-1} b c_2^{-1}$

Donc (S,T') reste connexe, et $\text{card } T' = \text{card } T = \text{card } S - 1$

Comme $\bar{a} \notin R$, on a $R \cup \{b\} \subseteq T'$

Par définition de b, $P(b) \leq P(\bar{a})$ (un que $\bar{a}$ et b ont une extrémité dans S_1 et l'autre dans S_2). Donc

$$\sum_{a \in T'} P(a) \geq \sum_{a \in T} P(a) - P(\bar{a}) + P(b) \leq \sum_{a \in T} P(a)$$

Comme (S,T) est un ACM, on déduit l'égalité et (S,T') est un ACM.

<u>Conséquence</u>. Si $R \subseteq T$ pour un ACM(S,T) et si C est une composante connexe de (S,R), et si b est une arête de poids minimum parmi celles reliant C à $S \setminus C$, alors $R \cup \{b\} \subseteq T'$ pour un ACM(S,T').

Algorithme de Kruskal

Trier les arêtes de G par ordre croissant de poids. Donc
$A = \{a_1, \dots, a_m\}$, avec $P(a_1) \leq \dots \leq P(a_m)$

$A_0 = \phi$

Pour $i := 1, \dots, n$ faire :

Si $\delta(a_i)$ n'est pas inclus dans une composante

Connexe de (S, A_{i-1})

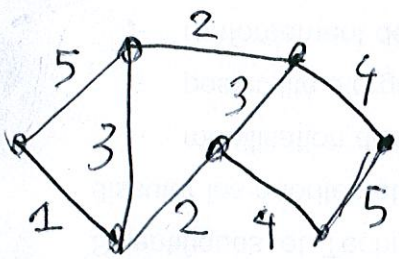
c-à-d si $A_{i-1} \cup \{a_i\}$ est sans cycle,

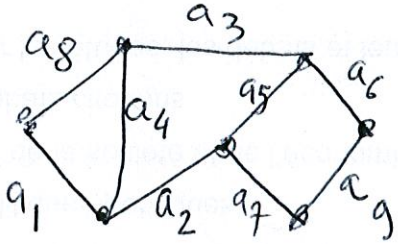
Alors $A_i = A_{i-1} \cup \{a_i\}$

Sinon $A_i = A_{i-1}$

Fin (S, A_n) est un ACM.

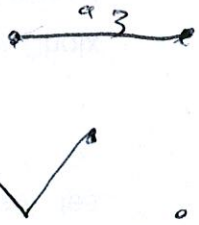
Donc on ajoute les arêtes par ordre croissant de poids, avec la seule contrainte de ne pas créer de cycle

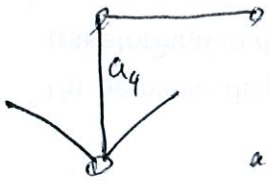


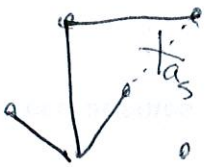


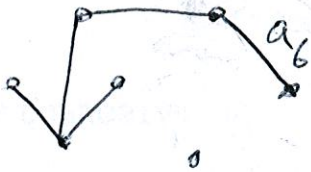
1) 

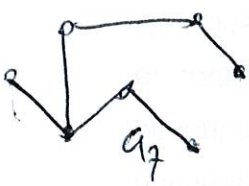
2) 

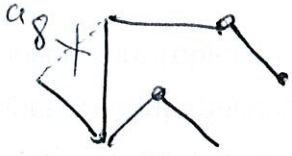
3) 

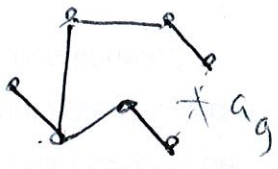
4) 

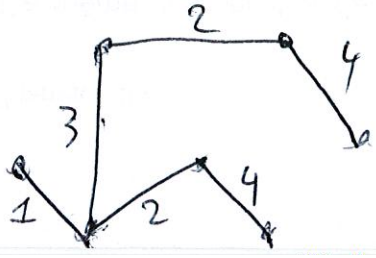
5) 

6) 

7) 

8) 

9) 

final 

Justification : Vu qu'à chaque étape on ~~ne peut~~ vérifie que la nouvelle arête ne crée pas de cycle (joint deux composantes connexes distinctes), on n'aura pas de cycle, et comme c'est l'arête de poids minimum (celles de poids moindre ont soit déjà été rajoutées, soit éliminées parce qu'elles créent un cycle), par la proposition la forêt A_i est un graphe partiel d'un ACM. Donc A_m est un graphe partiel d'un ACM. Si A_m n'était pas l'ACM, il serait non connexe, mais alors il y aurait une arête de poids minimum rajoutant des composantes, et elle aurait été ajoutée lors de l'étape i.

Algorithme de Prim. On va noter

un sous-graphe partiel qui est un arbre inclus dans un ACM. On part d'une racine r.

Soit $n = |S|$

$S_1 := \{r\}$, $A_1 := \emptyset$

pour $i = 2, \dots, n$, faire

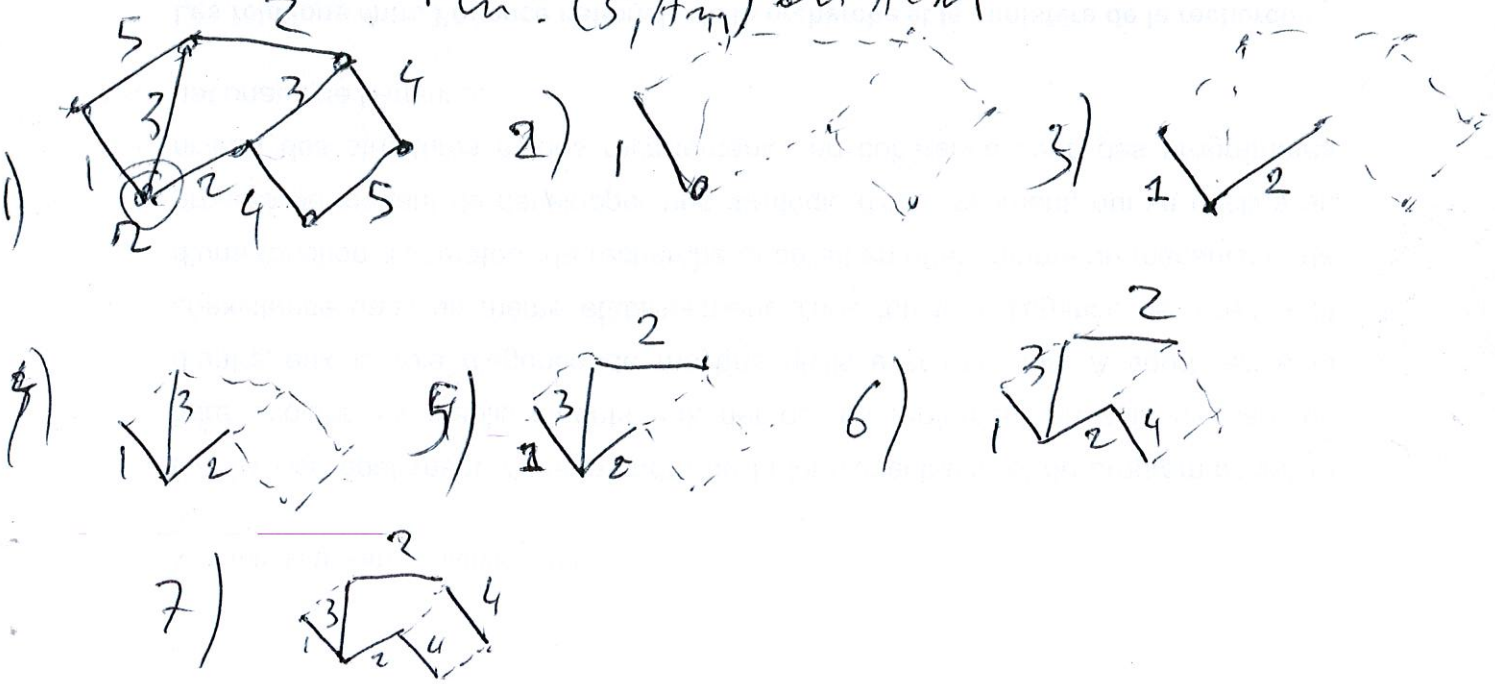
choisir parmi les arêtes ayant une extrémité dans S_{i-1} et l'autre dans $S \setminus S_{i-1}$, une de poids minimum : a

$S_i := S_{i-1} \cup \delta(a)$ (on rajoute à S_{i-1} l'extrémité de a qui est en dehors de S_i)

$A_i := A_{i-1} \cup \{a\}$

"Pour $\delta(a) = \{x, y\}$ avec $x \in S_{i-1}$ et $y \in S \setminus S_{i-1}$ on pose père$(y) := x$; ainsi l'arbre enraciné devient une arborescence.

Fin : (S, A_n) est ACM



Justification : à l'étape i, on a un arbre sur S_i. Le choix d'une arête joignant S_{i-1} à $S \setminus S_{i-1}$ de poids minimum garantit que (S, A_i) sera un graphe partiel d'un ACM. A la fin, on recouvre S, donc c'est un ACM

Algorithme par retrait d'arêtes :

en quelque sorte, c'est l'inverse de l'algorithme de Kruskal.

Trier les arêtes de G par ordre décroissant de poids. Donc

$A = \{a_1, \dots, a_m\}$, avec $p(a_1) \geq \dots \geq p(a_m)$

$A_0 = A$

Pour $i = 1, \dots, m$ faire :

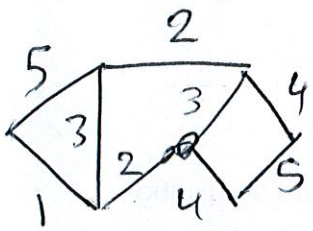
Si $A_{i-1} \setminus \{a_i\}$ est connexe, alors

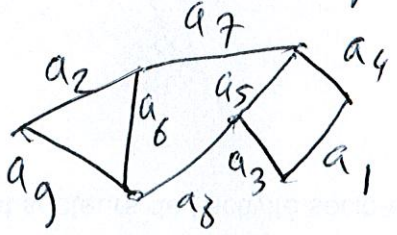
$A_i := A_{i-1} \setminus \{a_i\}$ sinon $A_i := A_{i-1}$

Fin. (S, A_m) est un ACM.

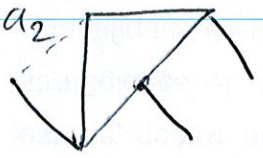
On enlève les arêtes par ordre décrois-

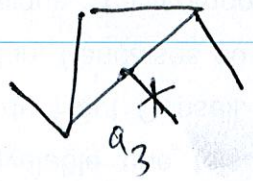
sant de poids, avec la seule contrainte de ne pas déconnecter le graphe.

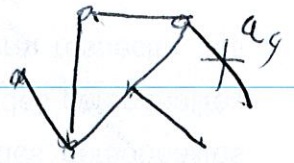


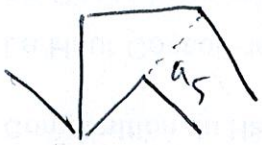


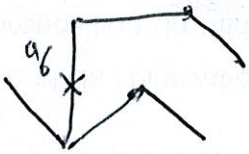
1) 

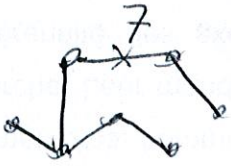
2) 

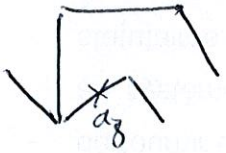
3) 

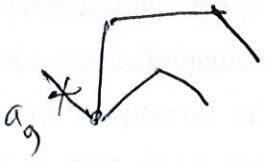
4) 

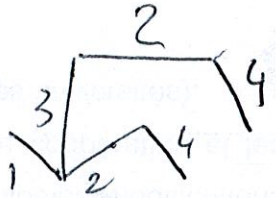
5) 

6) 

7) 

8) 



final 

Justification. (S, A_0) contient un ACM.

Supposons que $T \subseteq A_{i-1}$ pour un ACM (S, T)

Si $A_i = A_{i-1}$ ou si $a_i \notin T$, alors $T \subseteq A_i$

Supposons que $a_i \in T$ et $A_i = A_{i-1} \setminus \{a_i\}$, donc A_i est connexe. $T \setminus \{a_i\}$ a 2 composantes connexes C_1 et C_2, reliées par une arête $a_j \in A_i$! T. On a $j > i$, parce que si $j < i$, a_j aurait été enlevé à l'étape j ($T \subseteq A_{i-1} \subseteq A_{j-1} \setminus \{a_j\}$, donc connexe).

On prend $T' = (T \setminus \{a_i\} \cup \{a_j\})$, on a $T' \subseteq A_i$, et $\sum_{a \in T'} p(a) \leq \sum_{a \in T} p(a)$ (car $p(a_j) \leq p(a_i)$), donc (S, T') est un ACM.

Final A_n contient un ACM. Si A_n n'est pas l'ACM, il a un cycle, dans ce cycle soit a_i l'arête de poids maximum. Alors $A_n \setminus \{a_i\}$ est connexe, donc $A_{i-1} \setminus \{a_i\}$ l'est (car $A_n \subseteq A_{i-1}$), donc a_i devrait avoir été enlevé à l'étape i.

Propriété Si tous les poids des arêtes sont distincts, alors il n'y a qu'un seul ACM.

(Preuve en exercice).