Arbres recouvrants

Irena.Rusu@univ-nantes.fr
LINA, bureau 123, 02.51.12.58.16

Sommaire

- Graphes et sous-graphes
- Arbres recouvrants
- Arbres recouvrants de poids minimum
  - Méthode de Prim
  - Méthode de Kruskal

Sommaire

- Graphes et sous-graphes
- Arbres recouvrants
- Arbres recouvrants de poids minimum
  - Méthode de Prim
  - Méthode de Kruskal

Graphes

Intuitivement

Structure représentant des objets (les sommets) et des relations deux à deux, orientées ou non, entre les objets (arcs, arêtes)

Formellement

Graphe (orienté) $G = (S, A)$

S ensemble (fini) de sommets

$A \subseteq S \times S$ ensemble d' arcs, *i.e.*, relation sur S

Graphe non orienté $G = (S, A)$

$A \subseteq \mathrm{S} \times \mathrm{S}$ t.q. (x,y) et (y,x) sont identiques par convention

ensemble d'arêtes, relation symétrique

Note. Dans un cas plus général, plusieurs arc/arêtes peuvent exister entre deux sommets … on parle alors de (multi)graphe

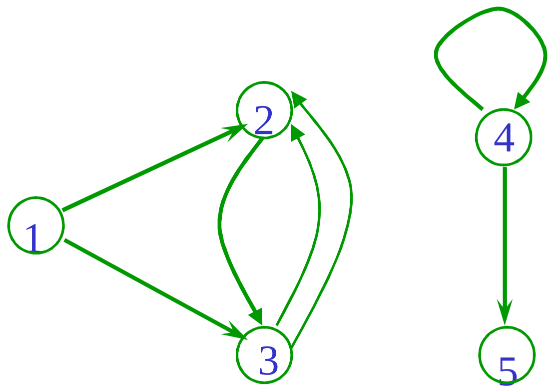
Graphes

Graphe (orienté) $G = (S, A)$

S ensemble (fini) des sommets

$A \subseteq S \times S$ ensemble des arcs,

i.e., relation sur S



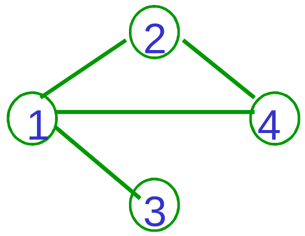
$S = \{ 1, 2, 3, 4, 5 \}$

$A = \{ (1, 2), (1, 3), (2, 3), (3, 2), (4, 4), (4, 5) \}$

Graphe non orienté $G = (S, A)$

$A \subseteq S \times S$ ensemble des arêtes,

relation symétrique

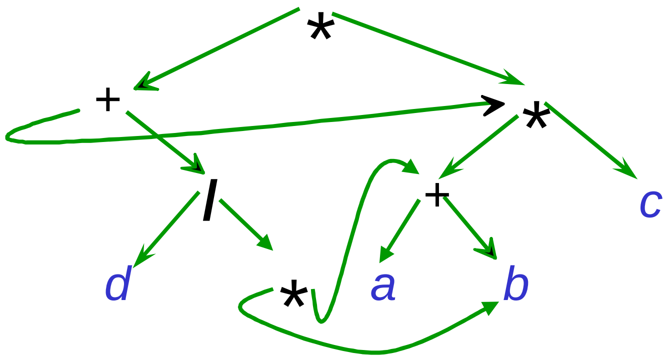


$S = \{ 1, 2, 3, 4 \}$

$A = \{ \{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 4\} \}$

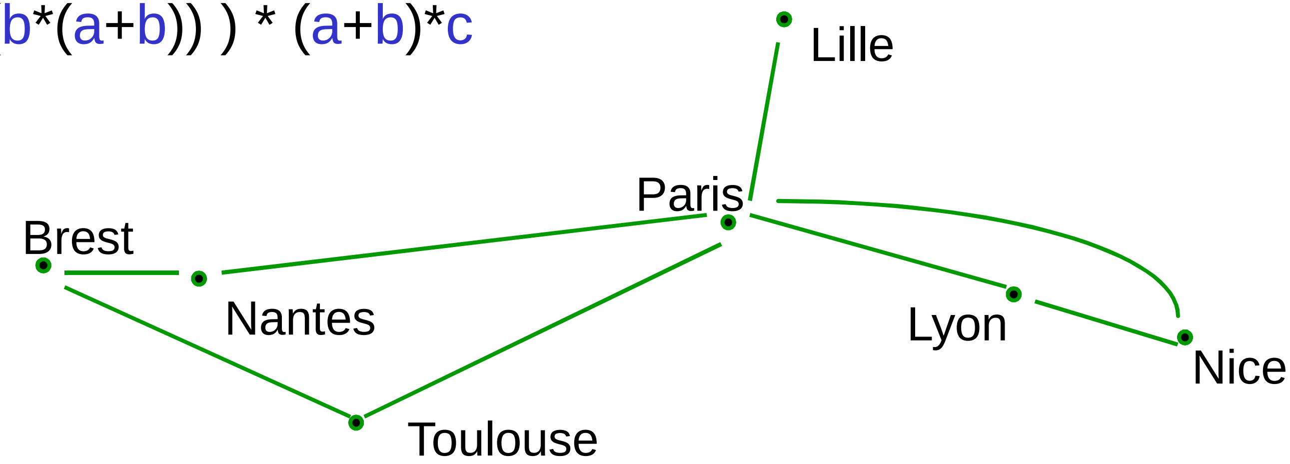
Exemples

Graphe acyclique d'une expression (DAG)



((a+b)*c+d/(b*(a+b))) * (a+b)*c

Réseau



Sous-graphes

G=(S,A) graphe non-orienté.

Un sous-graphe de G est un graphe G' dont tous les éléments (sommets, arêtes) sont inclus dans les éléments de G.

→ si (x,y) est une arête de G', alors x, y sont tous les deux des sommets de G'

→ si (x,y) est une arête de G', alors x, y sont des sommets de G et (x,y) est une arête de G.

Représentations des graphes en machine

$G = (S, A)$ $S = \{1, 2, ..., n\}$

Matrice d'adjacence

utilisation d'opérations matricielles

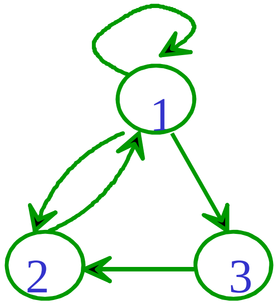
temps de traitement courant : quadratique

Listes de successeurs

réduit la taille si $|A| \ll |S|^2$

temps de traitement courant : $O(|S| + |A|)$

Matrices d'adjacence

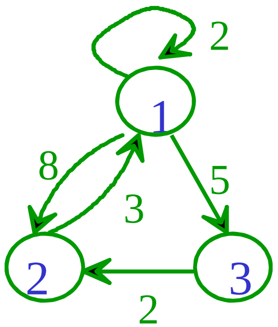


$S = \{ 1, 2, 3 \}$

$A = \{ (1,1), (1, 2), (1, 3), (2, 1), (3, 2)\}$

$$M = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

$M [i, j] = 1$ ssi j adjacent à i



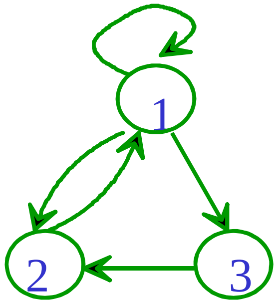
$$V = \begin{pmatrix} 2 & 8 & 5 \\ 3 & 0 & 0 \\ 0 & 2 & 0 \end{pmatrix}$$

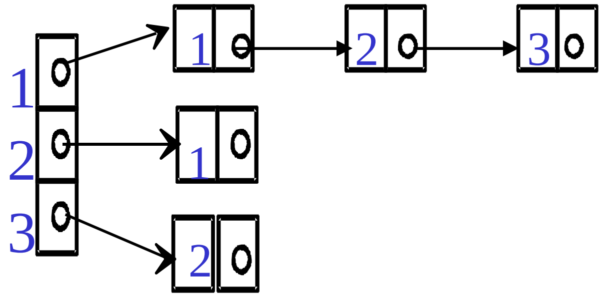
Valuation : $v : A \longrightarrow X$

Listes de successeurs

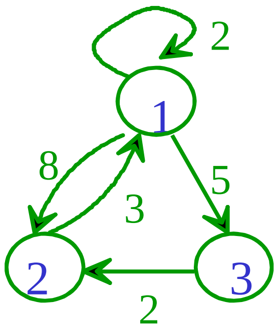
$S = \{ 1, 2, 3 \}$

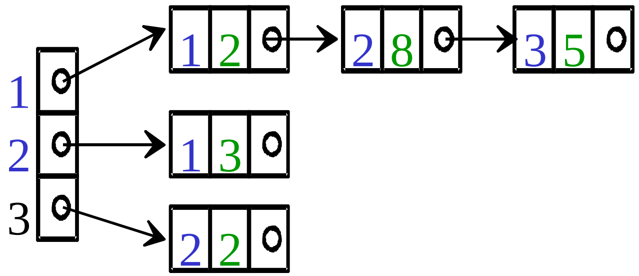
$A = \{ (1,1), (1, 2), (1, 3), (2, 1), (3, 2)\}$





Listes des $A(s)$





Valuation : $v : A \longrightarrow X$

Sommaire

- Graphes et sous-graphes
- Arbres recouvrants
- Arbres recouvrants de poids minimum
  - Méthode de Prim
  - Méthode de Kruskal

Arbres

- Un arbre T est un graphe :
  - S'il est vu comme une hiérarchie, c'est un graphe orienté
  - S'il n'est pas vu comme une hiérarchie, c'est un graphe non-orienté
- Dans ce chapitre, un arbre T n'est pas vu comme une hiérarchie

  → graphe non-orienté T.
- Ce qui le caractérise par rapport à un graphe arbitraire
  - Il est connecté ou connexe (toute paire de sommets est reliée par un « chemin » dans T)
  - Il n'a pas de cycles (il n'y a pas de chemin qui revienne à son point de départ)

Arbres recouvrants

Arbre recouvrant de G=(S,A) :

un arbre T=(S, A’) tel que A’ $\subseteq$ A

(autrement dit, un sous-graphe de G qui a les mêmes sommets et qui est un arbre)

Proposition.

Un graphe non-orienté admet un arbre recouvrant si et seulement si il est connexe.

Engendrer tous les arbres recouvrants d'un graphe

$G=(S,A)$ un graphe non-orienté

$S=\{1, 2, \ldots, n\}$, $|A|=m$

A donné par un tableau TA de taille $2xm$:

- **la *i*-ème arête de *G* est *{TA[1,i],TA[2,i]} avec TA[1,i]<TA[2,i]***
- **dans *TA*, les arêtes sont ordonnées par *TA[1,i]* croissant; on appelle *v* = *TA[1,1]* et *d(v)* le nombre d'arêtes de *TA* contenant *v*.**

Prem tableau, *i* entier tels que :

Dans l'arbre que l'on construit à un moment donné, les *i-1* premières arêtes sont

Prem[1]<Prem[2]<…<Prem[i-1]   (i=1, 2, …, n)

Algorithme

Procédure *ArbresRecouvrants* (*i* entier);

{engendre tous les arbres recouvrants de *G* dont les *i-1* premières arêtes sont les éléments d'indices *Prem[1], …, Prem[i-1]* du tableau TA}

si *i=n* **alors** fin

sinon si *i=1* **alors pour** *j←1* **à** *d(v)* **faire {**

Prem[i]←j;

ArbresRecouvrants(i+1)

}

sinon pour *j←Prem[i-1]+1* **à** *m* **faire**

si le sous-graphe de *G* défini par

Prem[1], …, Prem[i-1}, j est sans cycle simple

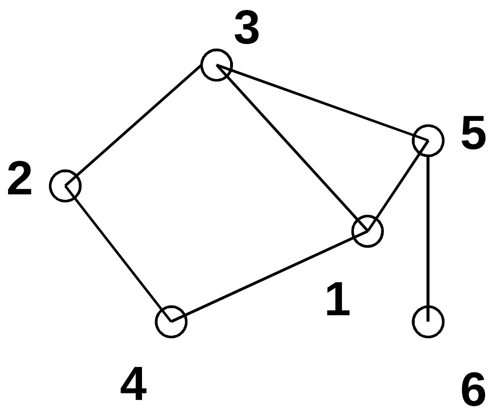
alors {

Prem[i]←j;

ArbresRecouvrants(i+1)

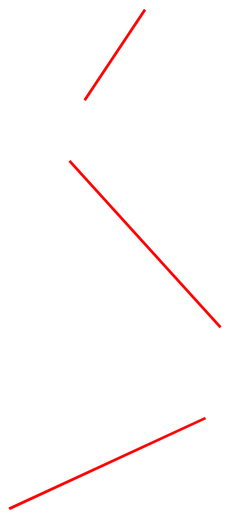
}

Exemple



TA :

1	2	3	4	5	6	7
1	1	1	2	2	3	5
5	3	4	3	4	5	6



i=1 **Prem**

1	2	3	4	5
1				

Appel récursif

i=2 **Prem**

2				

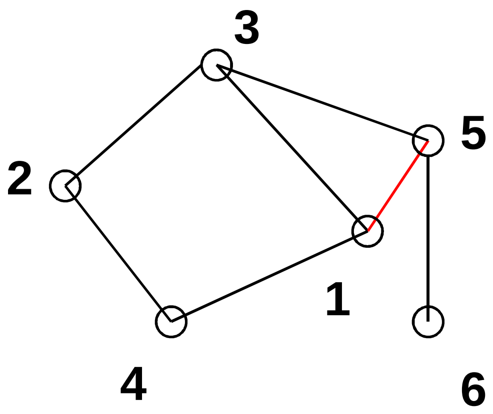
Appel récursif

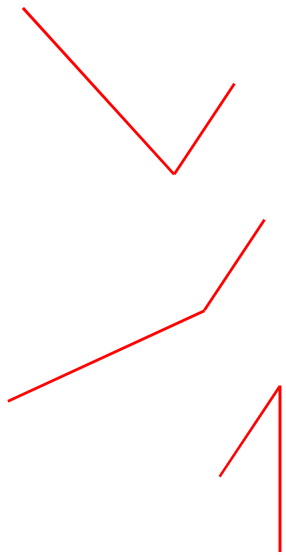
i=3 **Prem**

3				

Appel récursif

Exemple





	1	2	3	4	5	6	7
TA :	1	1	1	2	2	3	5
	5	3	4	3	4	5	6

i=1

	1	*2*	*3*	*4*	*5*
Prem	1				
j=2	1	2			

Appel récursif

j=3	1	3			

Appel récursif

…

j=7	1	7			

Appel récursif

17

Sommaire

- Graphes et sous-graphes
- Arbres recouvrants
- Arbres recouvrants de poids minimum
  - Méthode de Prim
  - Méthode de Kruskal

Arbres recouvrants de poids minimum

Calcul d’un arbre de poids minimum
recouvrant un graphe connexe

Applications : conception de réseaux
(téléphonique, électrique, d'intercommunication,...)
et étude de leur fonctionnement

Algorithmes
de Prim $O(n^2)$
(adapté aux matrices d’adjacence)
de Kruskal $O(m \log m)$
(adapté aux listes de successeurs
et graphes contenant peu d’arêtes)

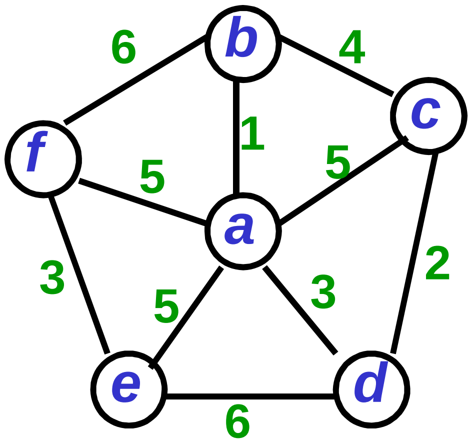
Le problème

Graphe valué $G = (S, A, v)$ avec valuation $v : A \rightarrow \mathbb{R}$
non-orienté et connexe

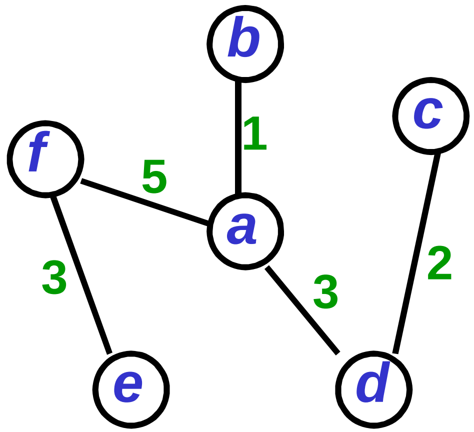
Poids (ou coût) d'un sous-graphe $G' = (S', A')$:

$$poids(G') = \Sigma\ (v(p,q) \mid (p,q) \in A')$$

Problème : déterminer un arbre recouvrant de poids minimum pour G



poids = 14

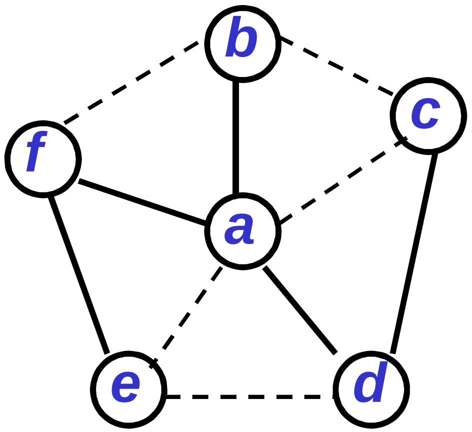


Propriétés

Graphe non-orienté et connexe $G = (S, A)$
$T = (S, B)$ arbre recouvrant pour G

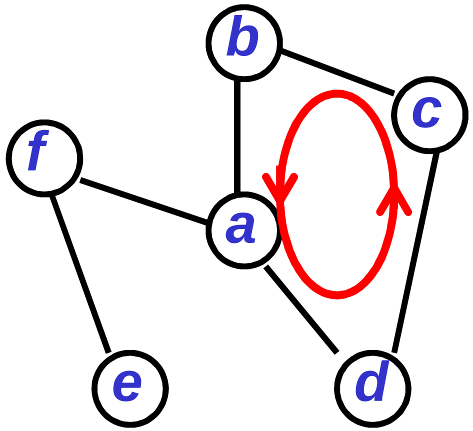
Propriétés

- T possède n-1 arêtes
- si $\{p, q\} \in A\text{-}B$ alors $H = (S, B + \{u,v\})$ possède un cycle



$|S| = 6$

$|B| = 5$



Sommaire

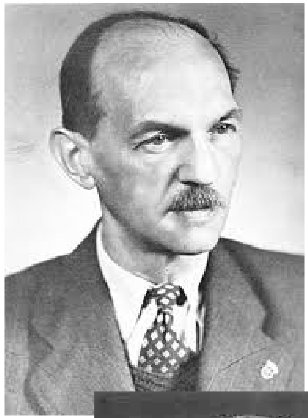
Algorithme de Prim : Qui, quoi, pourquoi ?

- Algorithme initialement proposé par

Vojtech Jarnik (1897-1970) en 1930.

About a certain minimal problem, Práce Moravské

Přírodovědecké Společnosti, 6, 1930, pp. 57–63. (in Czech).



- Retrouvé de manière indépendante par

Robert Prim (1921 -) en 1957:

Shortest connexion networks and some generalizations,

Bell System Technical Journal, 36: 1389-1401, 1957.

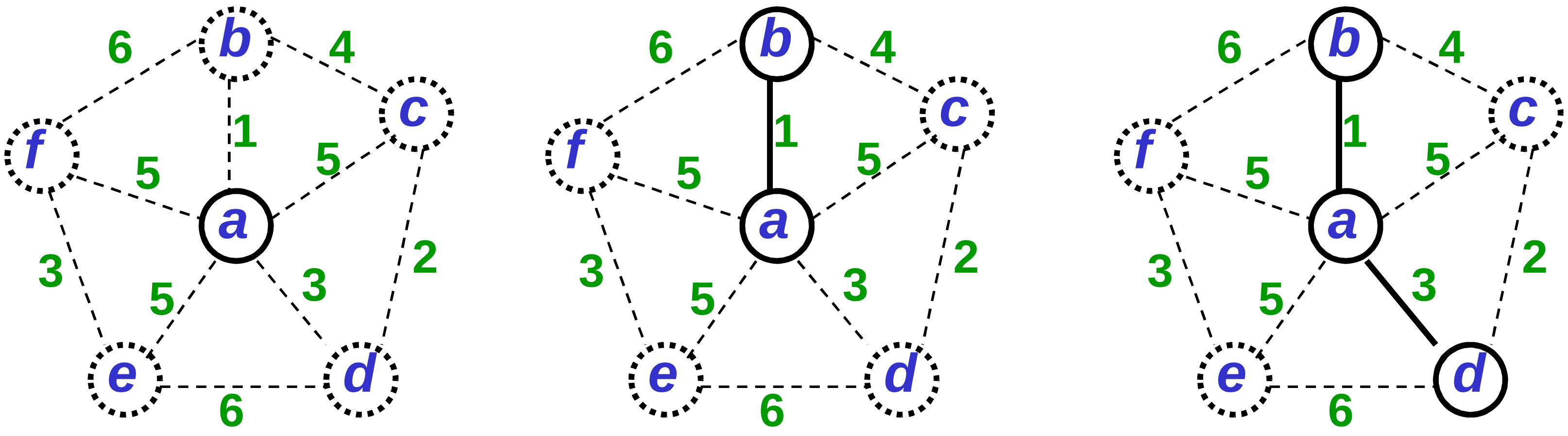
- Retrouvé indépendamment en 1959 par

Edsker Dikjstra (1930-2002)

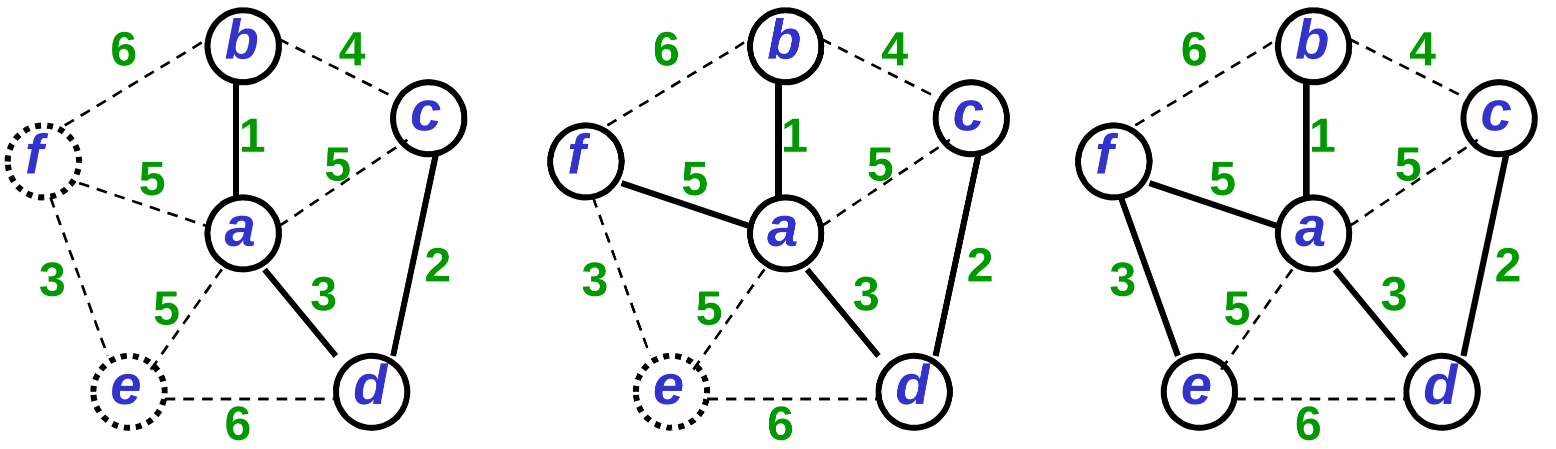
Méthode de Prim (1957) – algorithme glouton

Calcul d'un arbre recouvrant de poids minimum :

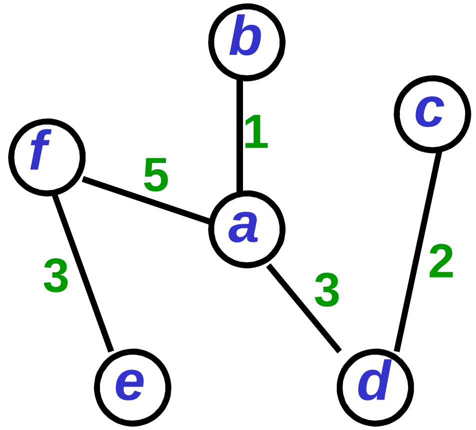
- **Commencer par un sommet**
- **Tant que le graphe n'est pas entièrement couvert faire**
    - **ajouter à l'arbre une arête de poids minimum**
    - **ajouter l'autre extrémité de cette arête**



Exemple (suite)



solution
poids = 14



Algorithme de Prim

Algorithme PRIM(graphe $(\{1, 2, ..., n\}, A, v)$) {
    $T \leftarrow \{1\}$;
    $B \leftarrow \emptyset$;
    tant que $|T| < n$ faire {
        $\{p, q\} \leftarrow$ arête de poids minimal
            telle que $p \in T$ et $q \notin T$;
        $T \leftarrow T + \{q\}$;
        $B \leftarrow B + \{p, q\}$;
    }
    retour (T, B) ;
}

Temps d'exécution : $O(n^2)$ au moyen de deux tables indexées par les sommets

Implémentation

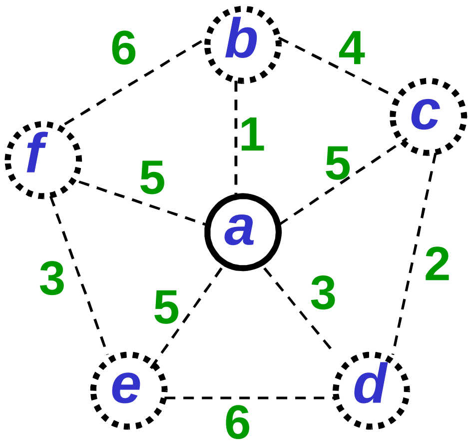
Tables `proche` et `poids` pour trouver l'arête $\{p, q\}$

$q \notin T$ `proche[`q`]` $= p \in T$

ssi $v(p, q) = \min \{ v(p', q) \mid p' \in T \}$

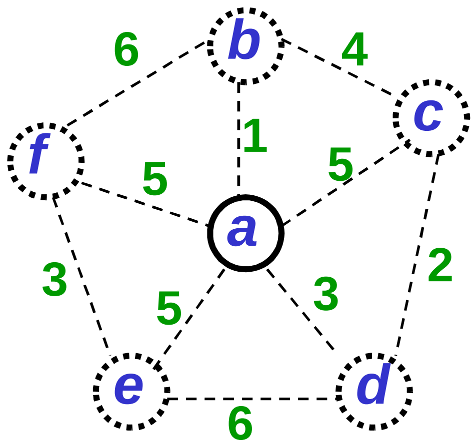
$q \notin T$ `poids[`q`]` $= v($`proche[`q`]`$, q)$

$q \in T$ `poids[`q`]` $= +\infty$



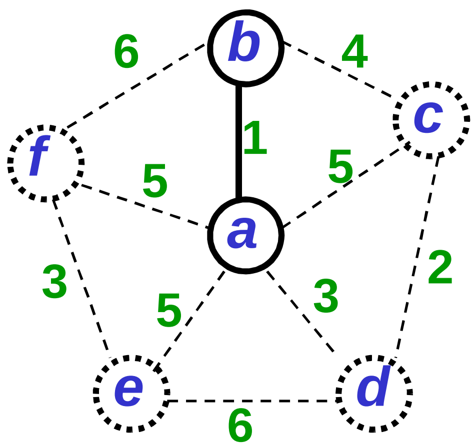
	a	b	c	d	e	f
`proche`		a	a	a	a	a
`poids`	∞	1	5	3	5	5

Une étape



	a	b	c	d	e	f
proche		a	a	a	a	a
poids	∞	1	5	3	5	5

ajout de *b* et {*a*, *b*}



	a	b	c	d	e	f
proche		a	b	a	a	a
poids	∞	∞	4	3	5	5

Temps O(1+|*A*(*b*)|)

Sommaire – Partie Arbres

- Généralités sur les arbres
- Arbres recouvrants
- Arbres recouvrants de poids minimum
  - Méthode de Prim
  - Méthode de Kruskal

Algorithme de Kruskal – Historique

- Joseph Kruskal (1928-2010)
- Collègue de R. Prim aux Laboratoires Bell.

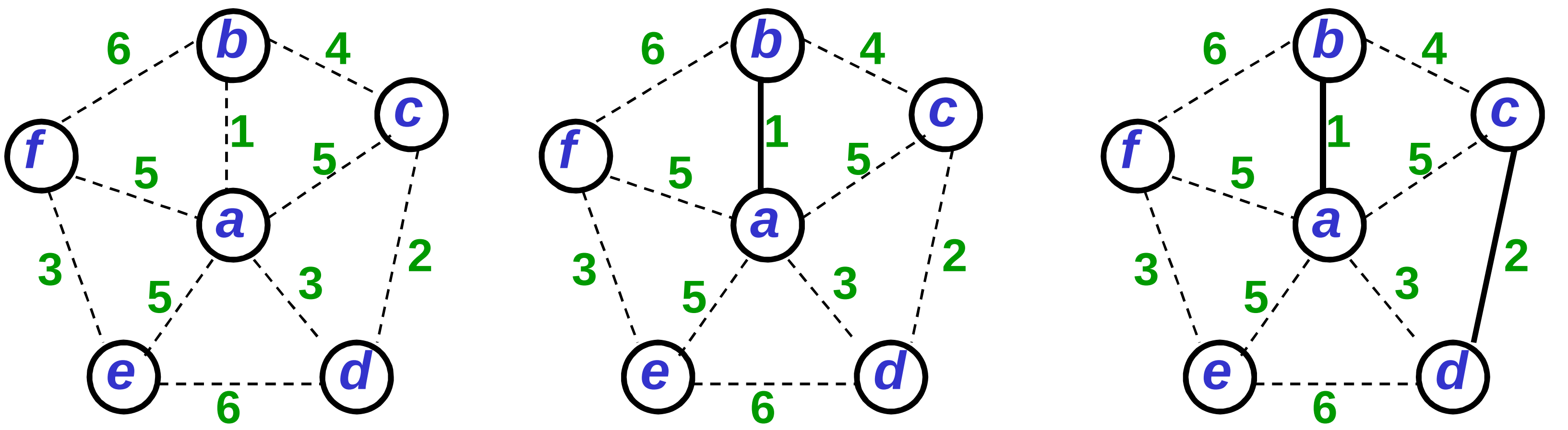
- Publication en 1956 :

On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem. In: *Proceedings of the American Mathematical Society*, Vol 7, No. 1 (Feb, 1956), pp. 48–50

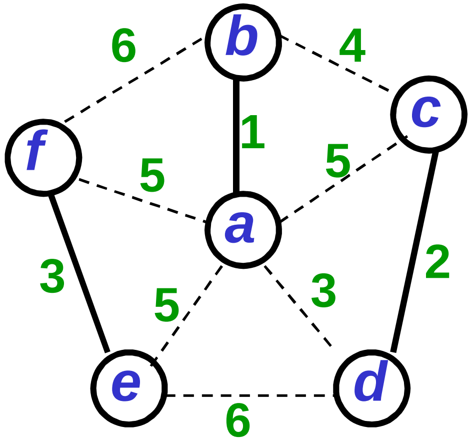
Méthode de Kruskal (1956)

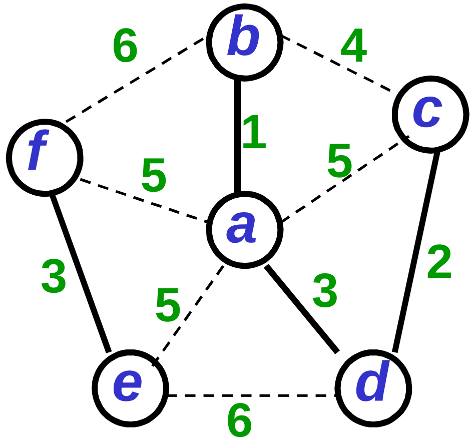
Calcul d'un arbre recouvrant de poids minimum :

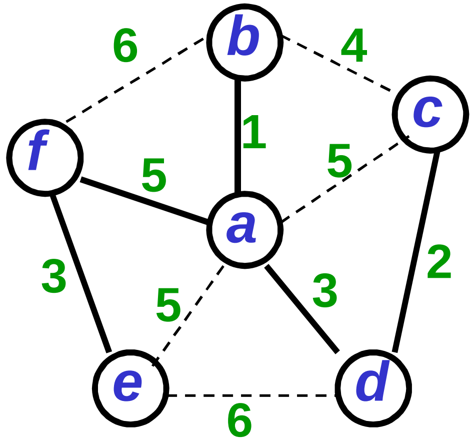
- Commencer par n sous-arbres (= sommets isolés)
- Tant qu'il reste au moins deux arbres faire
  - réunir deux sous-arbres disjoints par une arête de poids minimal



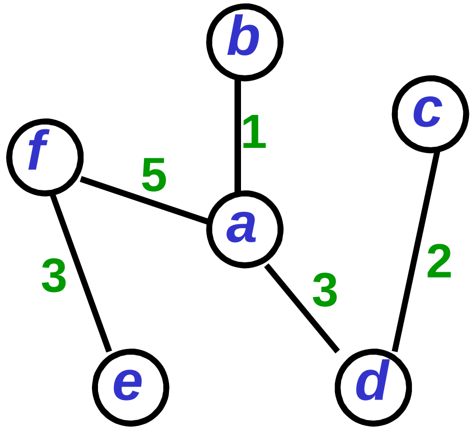
Exemple (suite)







solution
poids = 14



Algorithme de Kruskal

Algorithme KRUSKAL(graphe ({1, 2, ..., *n*}, *A*, *v*)) {

liste $\leftarrow$ *arêtes de A par ordre croissant des poids*;

$B \leftarrow \emptyset$;

tant que $|B| < n - 1$ faire {

soit $\{p, q\}$ le premier élément de *liste* ;

liste $\leftarrow$ *liste* - $\{p, q\}$;

si le graphe ({1, 2, ..., *n*}, $B \cup \{p, q\}$) est sans cycles alors

$B \leftarrow B \cup \{p, q\}$

}

retour ({1, 2, ..., *n*}, *B*) ;

}

Temps d'exécution : $O(m \log m)$ utilisant des méthodes de type CLASSE/UNION

Et après ?

- L'algorithmique des graphes en général et des arbres en particulier va beaucoup plus loin.

- Les problèmes à traiter sont de moins en moins évidents …

- … et ont de plus en plus besoin
  - d'expérience
  - de réflexion
  - et même d'acharnement

  pour les résoudre, ainsi que de preuves pour convaincre.

- Si ça vous dit d'aller plus loin … **master ORO**.