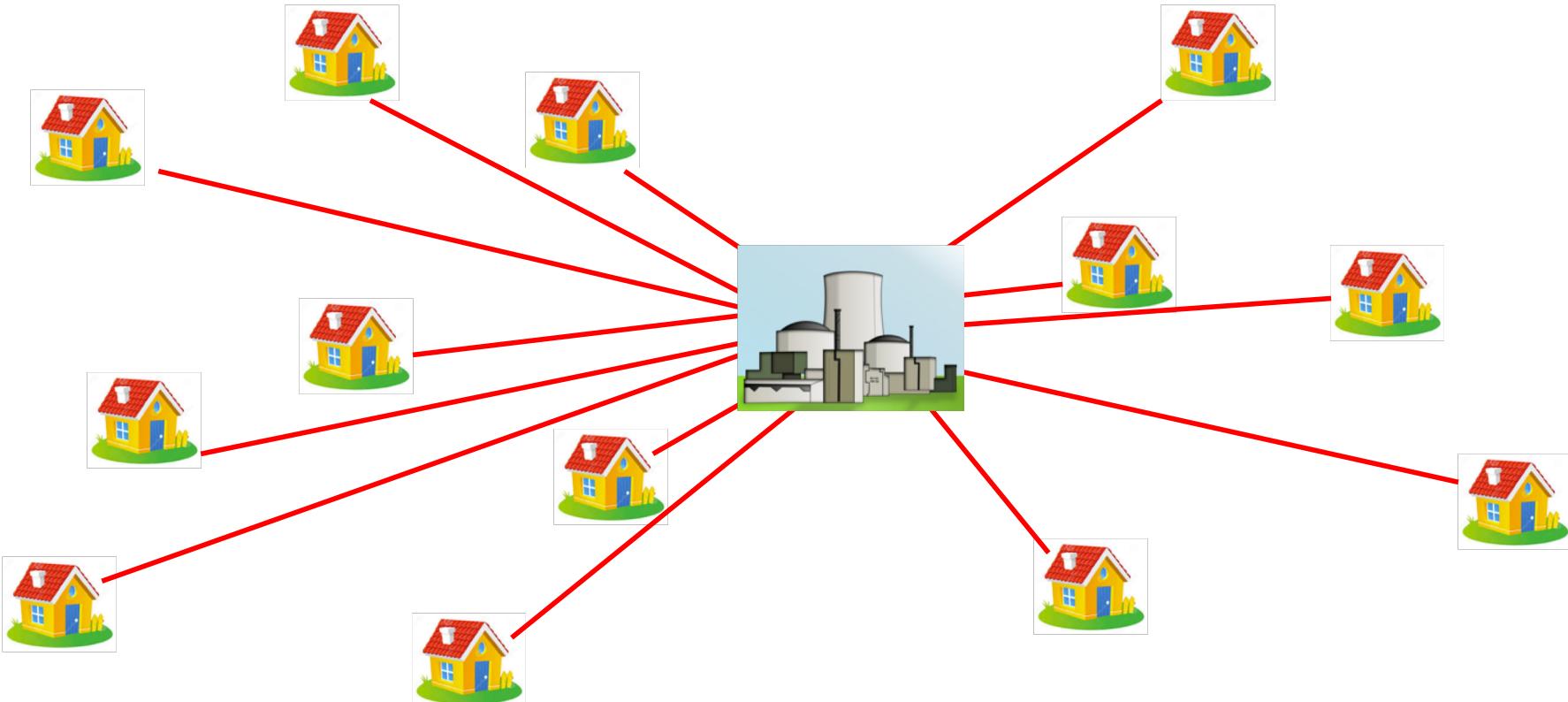


PLAN

1. Introduction
2. Généralités sur les graphes
3. Représentation d'un graphe en machine
4. Parcours dans les graphes
- 5. Arbre recouvrant**
6. Plus court chemin dans un graphe
7. Coloration d'un graphe
8. Graphes planaires
9. Flots et réseaux de transports
10. Analyse des réseaux d'interactions

CONSTRUCTION D'UN RÉSEAU ÉLECTRIQUE

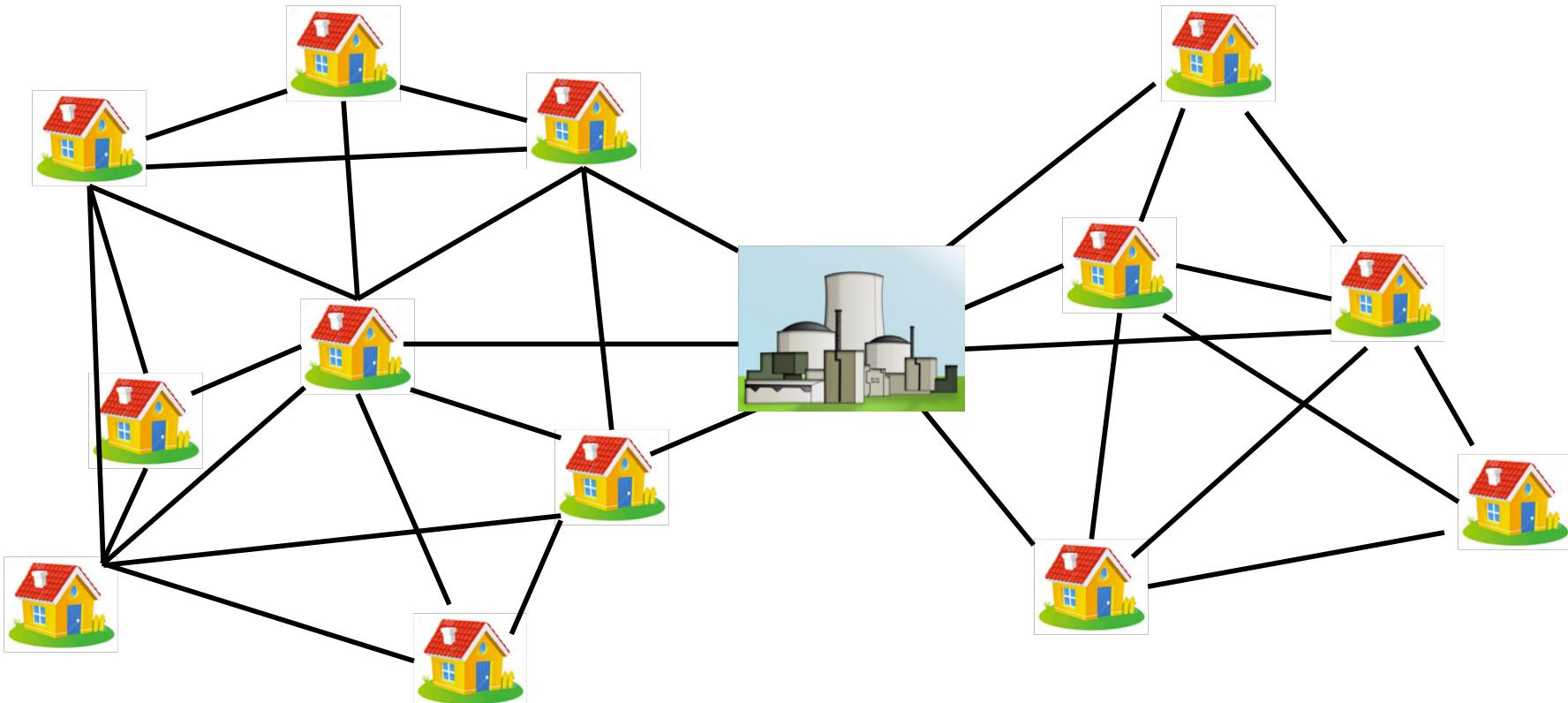
Construire un réseau électrique de manière à alimenter l'ensemble des maisons en électricité. Les coûts de construction des lignes sont proportionnels aux distances.



Comment construire le réseau électrique de manière à minimiser les coûts de fabrication ?

CONSTRUCTION D'UN RÉSEAU ÉLECTRIQUE

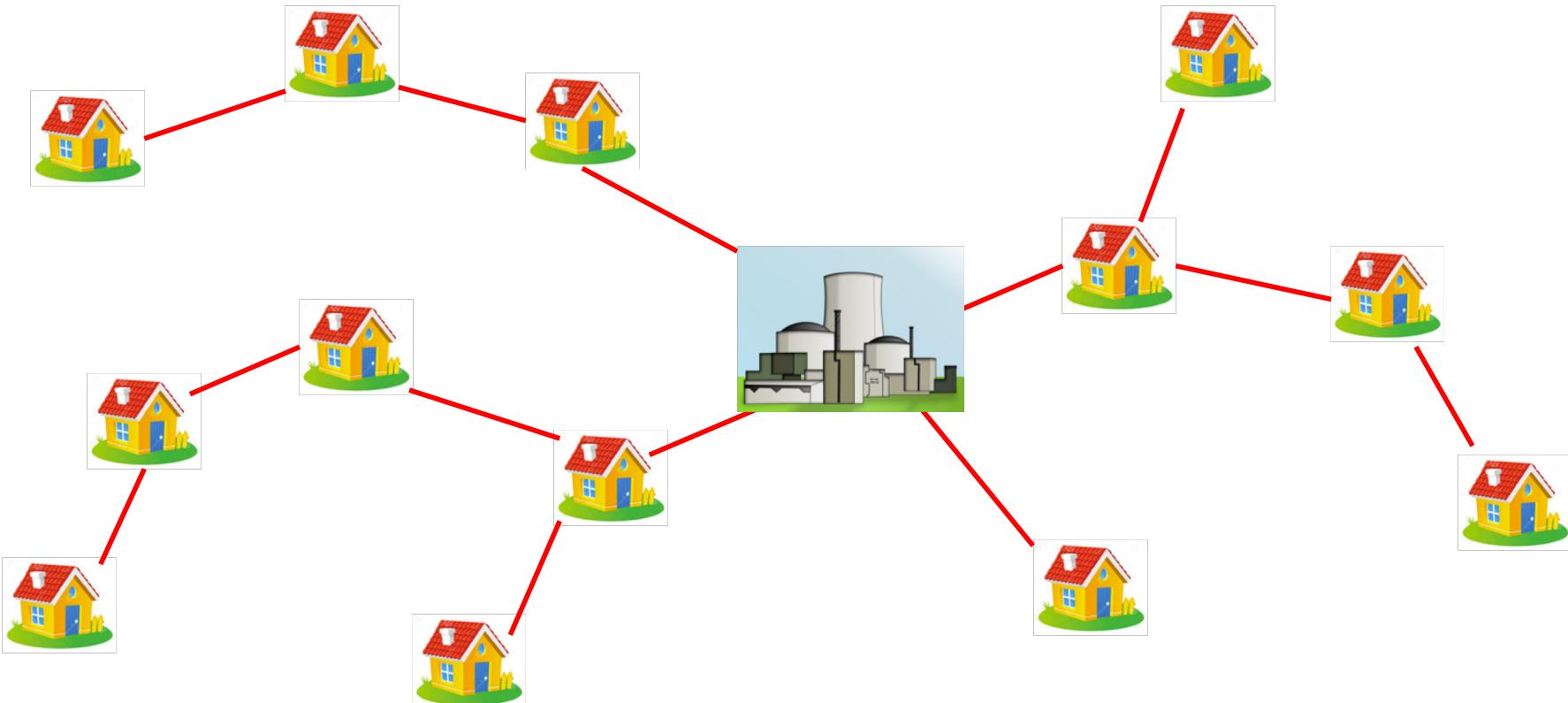
Graphe de « voisinage » non orienté :

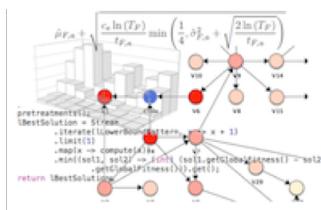


Recherche d'un arbre recouvrant de poids minimum

CONSTRUCTION D'UN RÉSEAU ÉLECTRIQUE

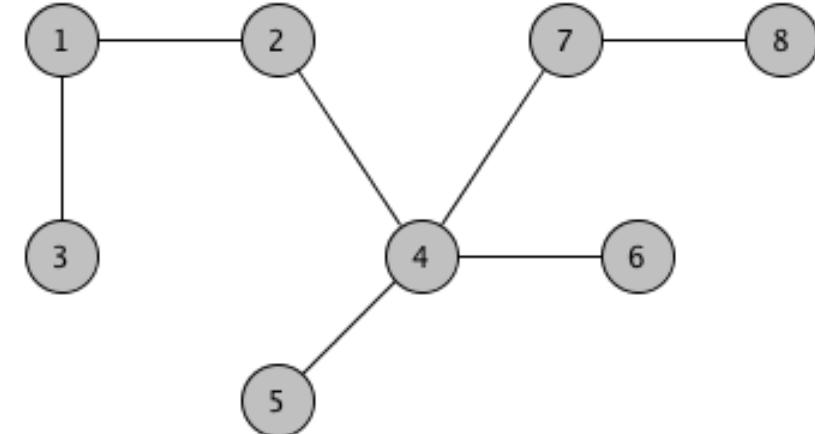
Arbre recouvrant de poids minimum :



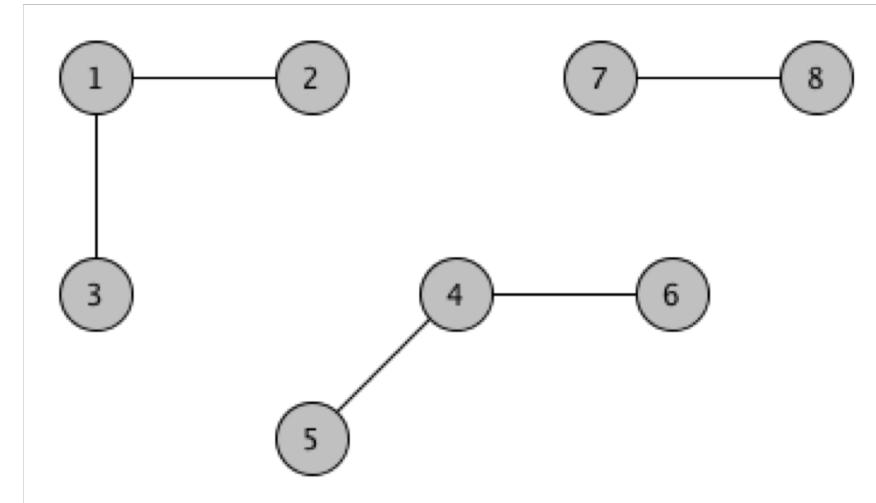


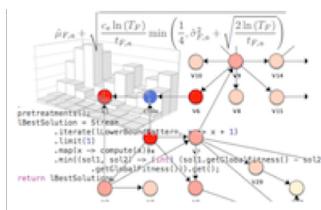
ARBRE ET FORÊT

Un **arbre** est un graphe connexe sans cycle (orienté ou non)



On appelle **forêt** un graphe dont chaque composante connexe est un arbre





ARBRE

Soit $G=(X,U)$ un graphe (avec $n \geq 2$).

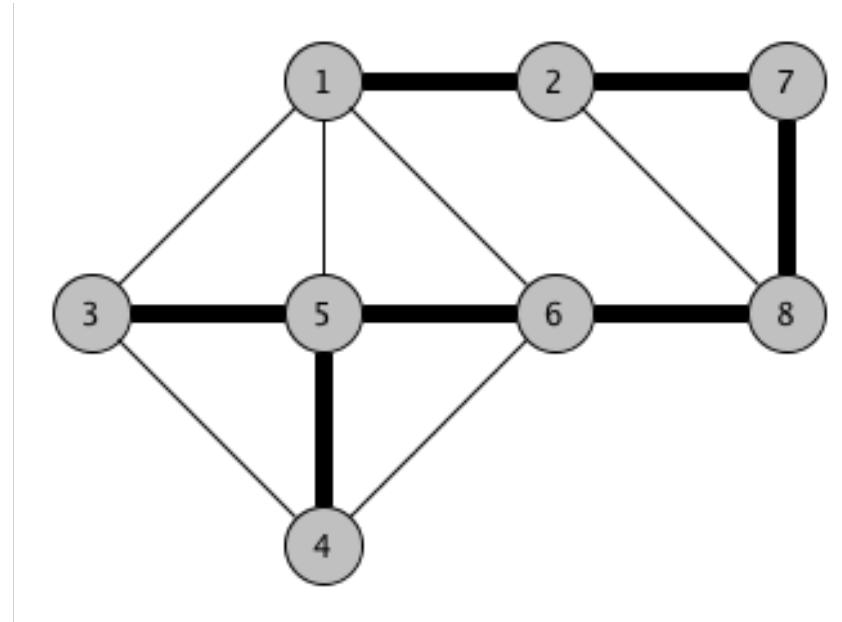
Les propositions suivantes sont équivalentes :

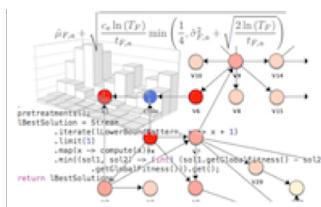
1. G est un arbre
2. Il existe dans G une chaîne et une seule joignant tout couple de sommets
3. G est sans cycle et maximal pour la propriété : « si on ajoute une arête, G possède un cycle »
4. G est connexe et minimal pour la propriété : « si on supprime une arête, G n'est plus connexe »
5. G est sans cycle et possède $(n-1)$ arêtes
6. G est connexe et possède $(n-1)$ arêtes

ARBRE

Théorème

Tout graphe G connexe possède un graphe partiel qui est un arbre



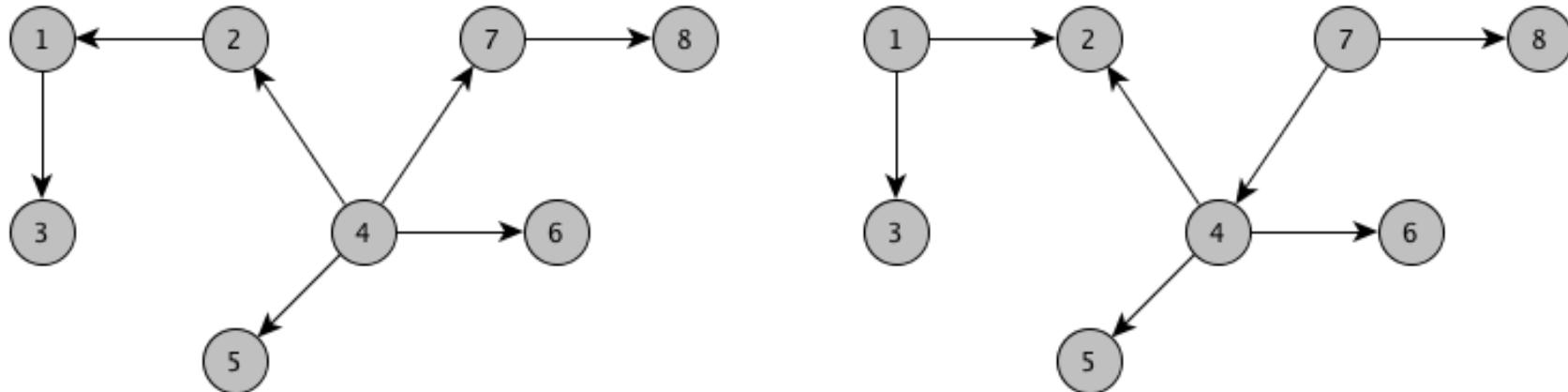


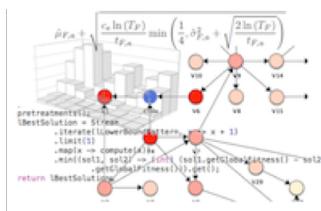
ARBORESCENCE

Un sommet s d'un graphe orienté G est une **racine** s'il existe dans G un chemin joignant s à tous les sommets de X

Un sommet s d'un graphe orienté G est une **anti-racine** s'il existe dans G un chemin joignant tous les sommets de X à s

Un graphe orienté G est une **arborescence** de racine s si G est un arbre et s une racine de G

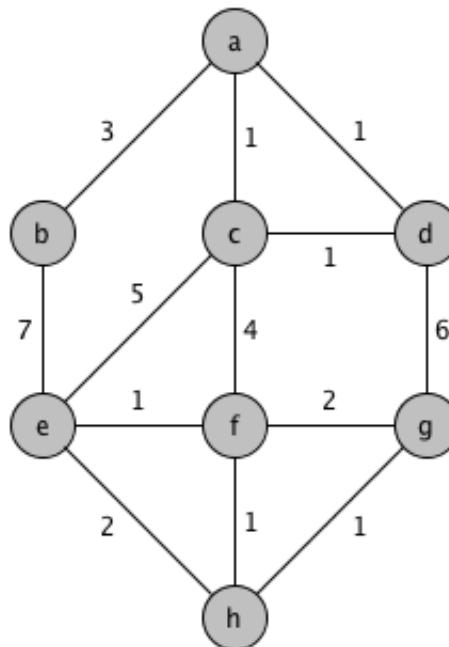




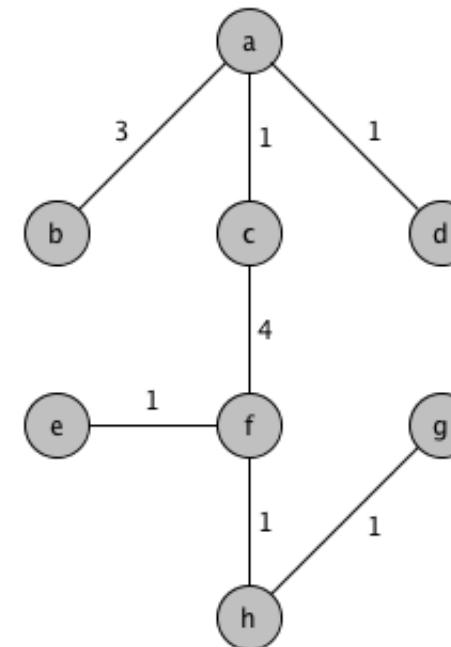
ARBRE RECOUVRANT DE POIDS MINIMUM (OU MAXIMUM)

Objectif

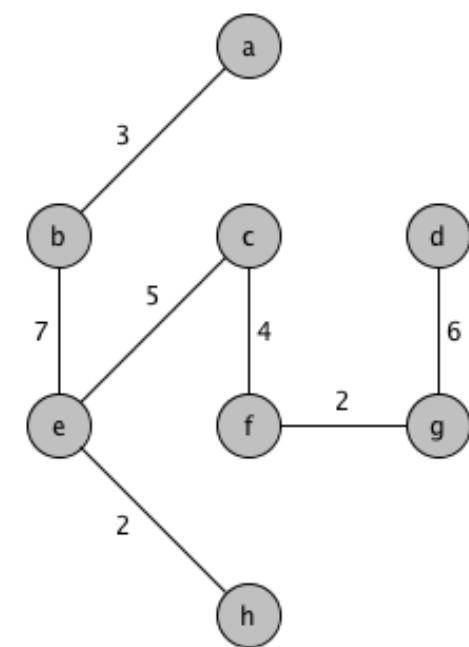
Trouver un graphe partiel d'un graphe valué $G=(X,U,I)$ qui soit un arbre de poids minimum (ou maximum)



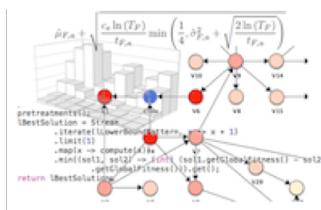
G



T_{min}



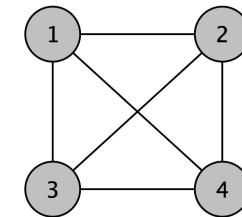
T_{max}



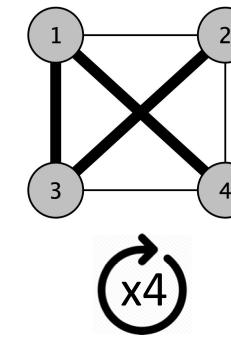
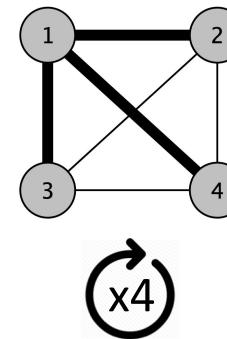
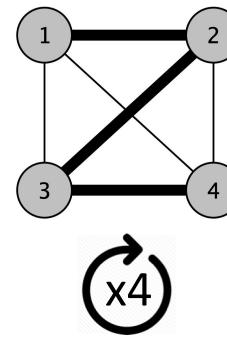
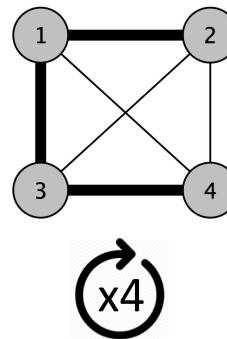
ARBRE RECOUVRANT DE POIDS MINIMUM (OU MAXIMUM)

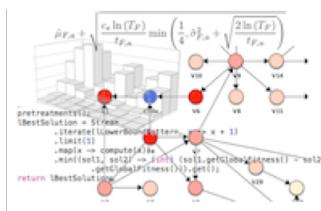
Le nombre d'arbres recouvrants peut être important, même sur des petits graphes.

Lister tous les graphes partiels du graphe non orienté complet à 4 sommets (K_4) qui sont des arbres :



Il y a 16 arbres recouvrants différents :



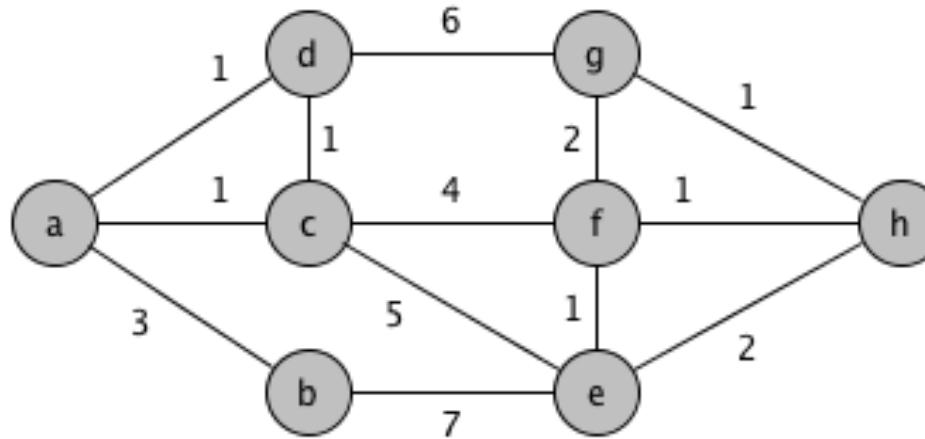


ALGORITHME DE KRUSKAL - VERSION 1

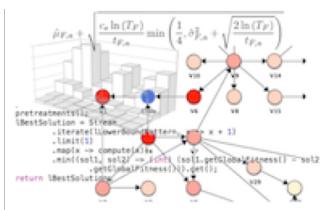
```

Kruskal1(G=(X,U,I),T) ; // recherche d'un arbre recouvrant de poids minimum
{
    Numéroter les arêtes dans l'ordre des poids croissants ( $I(u_1) \leq I(u_2) \leq \dots \leq I(u_m)$ ) ;
    T =  $\emptyset$  ;
    i = 0 ;
    tant que  $|T| < n-1$  faire
    {
        si  $(X, T \cup \{u_i\})$  ne contient pas de cycle faire  $T = T \cup \{u_i\}$  ;
         $i=i+1$  ;
    }
}

```



$O(m \log(m))$



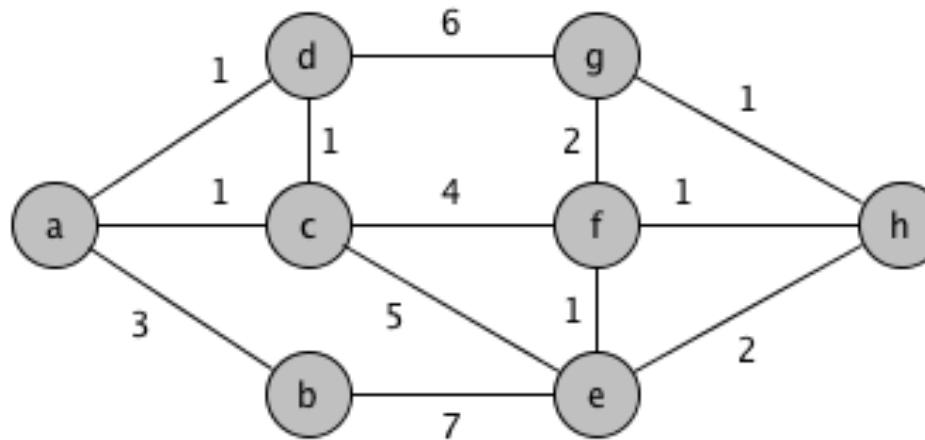
ARBRE RECOUVRANT

ALGORITHME DE KRUSKAL - VERSION 2

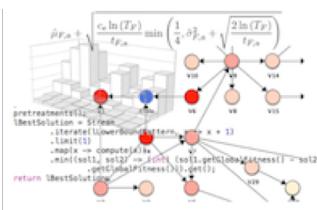
```

Kruskal2(G=(X,U,I),T) ; // recherche d'un arbre recouvrant de poids minimum
{
    Numéroter les arêtes dans l'ordre des poids décroissants ( $I(u_1) \geq \dots \geq I(u_m)$ ) ;
    T = U ;
    i = 0 ;
    tant que  $|T| \geq n$  faire
    {
        si  $(X, T - \{u_i\})$  est connexe faire  $T = T - \{u_i\}$  ;
         $i = i + 1$  ;
    }
}

```



$O(m \log(m))$



ALGORITHME DE PRIM

Prim(G=(X,U,I),T) ;

{ prendre un sommet x quelconque dans X ;

R = {x} ;

T = \emptyset ;

Tant que X \neq R faire

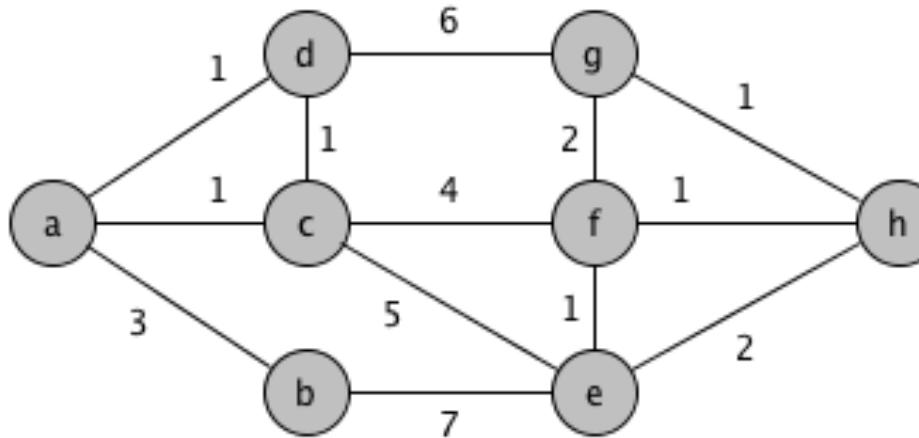
{ prendre (y,z) l'arête de poids minimum tel que y \in R et z \in X-R ;

T = T \cup {(y,z)} ;

R = R \cup {z} ;

}

}



O(n²)