Introduction à la Théorie des Graphes

--

Chap 4 : Arbres couvrants de poids minimal

Dr. Guelzim ibrahim

Email: ib.guelzim@gmail.com

Sommaire

Arbres et arbres couvrants

- Rappel
  - Un graphe G est connexe si pour toute paire {x, y} de sommets de G il existe une chaîne reliant x et y.
  - Une chaîne ou un cycle sont simples s'ils ne passent pas deux fois par la même arête.
  - Cette notion ne fait pas intervenir l'éventuelle orientation du graphe.
- Définition
  - Soit G= (V,E) un graphe orienté ou non.
  - On dit que G est un **<u>arbre</u>** SSI G est connexe et ne possède pas de cycles simples.

Arbres et arbres couvrants

- Définition équivalente
  - Soit $G = (V, E)$ un graphe orienté ou non.

  - On dit que G est un **<u>arbre</u>** SSI pour toute paire $\{x, y\}$ de sommets de G, il existe une **<u>unique chaîne simple</u>** d'extrémités x et y.

Arbres et arbres couvrants

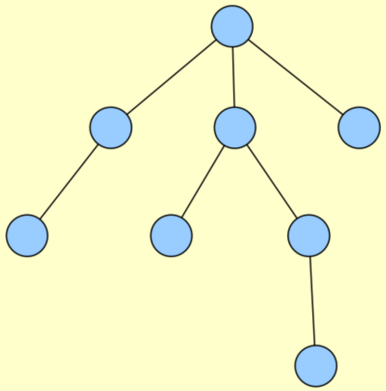
- Démonstration de l'équivalence des deux définitions d'un arbre
  - Considérons un graphe G connexe et sans cycles simples, et fixons une paire de sommets {x , y}.
  - Par définition de la notion de connexité, il existe une chaîne d'extrémités x et y.
  - En retirant éventuellement des arêtes de cette chaîne on peut même la rendre simple.
  - Il reste à vérifier qu'elle est unique.
  - Raisonnons par l'absurde en supposant qu'il existe au moins une autre.
  - On a alors deux chaînes simples différentes reliant x et y, (x, a1, a2, ..., an, y) et (x, b1, b2, ..., bm, y).
  - Leur concaténation donne alors (x, a1, a2, ..., an, y, bm, ..., b2, b1, x),
  - i.e. un cycle simple, ce qui est absurde car le graphe G n'en contient pas.

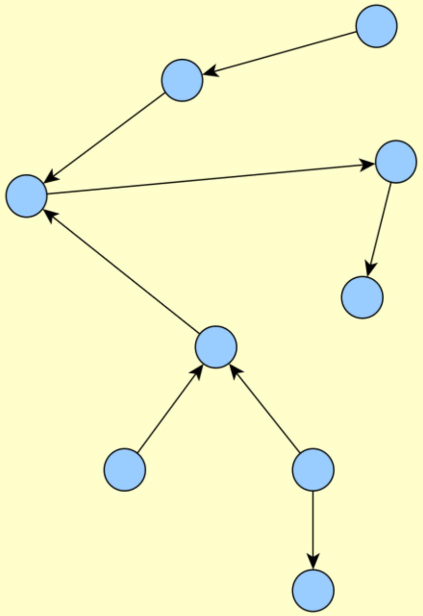
Arbres et arbres couvrants

- Démonstration de l'équivalence des deux définitions d'un arbre
  - La réciproque utilise le même genre d'arguments.
  - Prenons un graphe G dans lequel entre deux sommets quelconques il existe une unique chaîne simple.
  - Raisonnons là aussi par l'absurde en supposant que G possède un cycle simple.
  - Soit x l'origine et extrémité de ce cycle, et soit y un sommet arbitrairement choisi à l'intérieur.
  - Ce cycle est donc de la forme (x, a1, a2, ..., an, y, bm, ..., b2, b1, x).
  - Cela entraîne l'existence de deux chaînes simples d'extrémités x et y, à savoir (x, a1, a2, ..., an, y) et (x, b1, b2, ..., bm, y) , ce qui est absurde.

Arbres et arbres couvrants

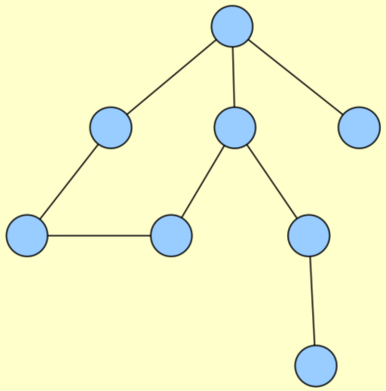
- Exemples:
  - Ces deux graphes vérifient bien les contraintes de la définition d'arbre,
  - Ils sont connexes et ne possèdent pas de cycles simples.

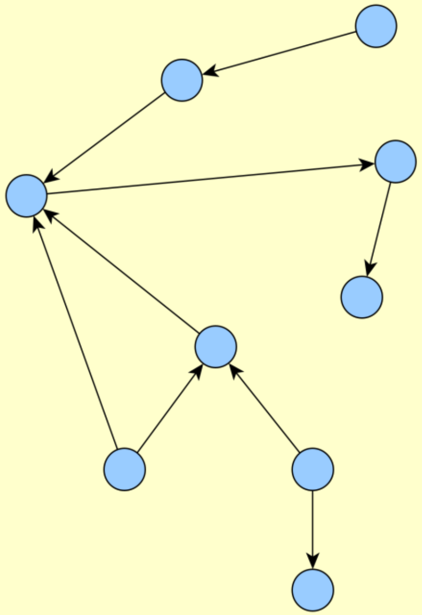




Arbres et arbres couvrants

- Exemples
  - Les deux graphes suivants contiennent des cycles simples,
  - Ce ne sont donc pas des arbres





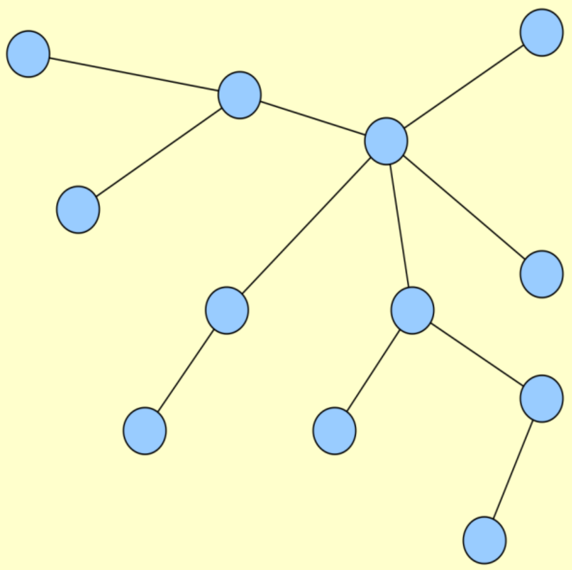
Arbres et arbres couvrants

- Rappel :
  - L'ordre d'un graphe est son nombre de sommets.

- Propriétés
  - Soit $G = (V, E)$ un graphe non orienté d'ordre n. Les propositions suivantes sont équivalentes :
    - G est un arbre.
    - G est connexe et possède $n - 1$ arêtes.
    - G est sans cycles simples et possède $n - 1$ arêtes

Arbres et arbres couvrants

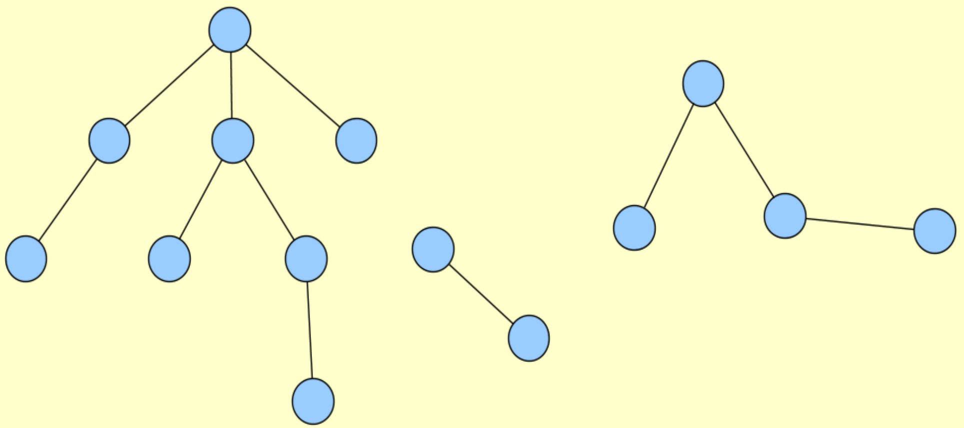
- Le graphe ci dessous est un arbre, il possède bien 12 sommets et 11 arêtes



- Remarques :
  - Si le graphe en comporte moins il ne pourra pas être connexe
  - S'il en possède plus il aura nécessairement des cycles simples.

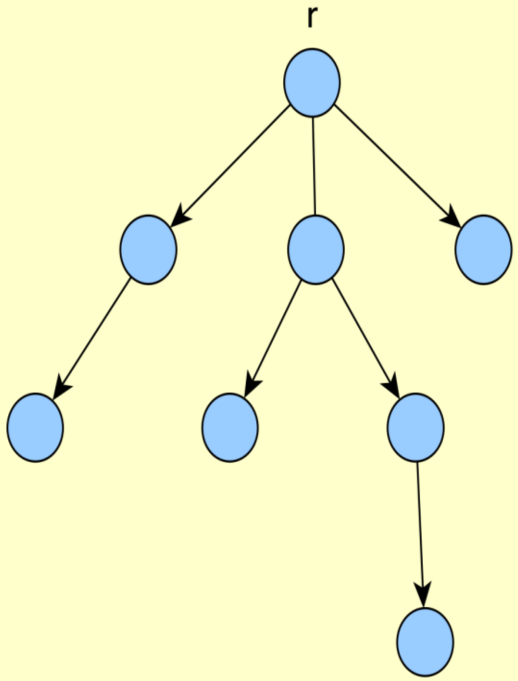
Arbres et arbres couvrants

- Définition
  - Soit G = (V , E) un graphe orienté ou non.
  - On dit que G est une **forêt** SSI G n'est pas connexe et ne possède pas de cycles simples.
- Les composantes connexes de G sont des arbres.
- Exemple :
  - Le graphe ci-contre est une forêt,
  - Chacune de ses composantes connexes étant un arbre



Arbres et arbres couvrants

- Définitions:
  - Dans un graphe **orienté**, une **racine** est un sommet particulier, tel que pour tout autre sommet x, il existe un **unique chemin simple** d'origine r et d'extrémité x .
  - Un graphe G est une **arborescence** si G est un arbre orienté muni d'une racine,
- Exemple:
  - Le graphe ci-contre est une arborescence de racine r



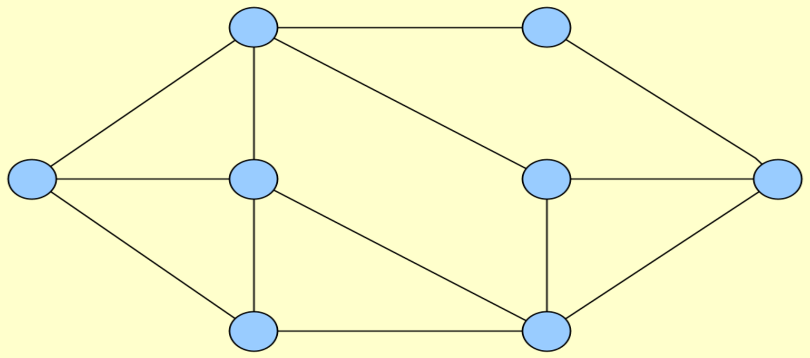
Arbres et arbres couvrants

- Définitions:
  - Un sous-graphe **couvrant** d'un graphe *G* est un sous-graphe de *G* contenant tous ses sommets.
  - Quand un arbre couvrant existe, il n'est pas nécessairement unique.

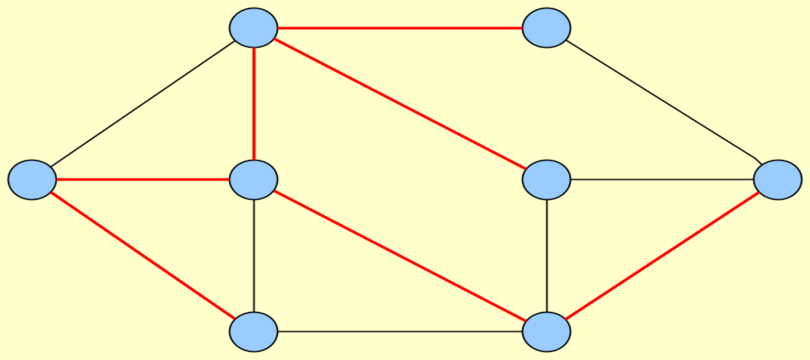
  - Soit $G = (V, E)$ un graphe non orienté.
  - Un **arbre couvrant** de *G* est un sous-graphe couvrant de *G* qui est un arbre.
  - i.e. : sous-graphe couvrant de *G* qui est à la fois connexe et sans cycles simples.

Arbres et arbres couvrants

- Considérons le graphe G non orienté suivant



- Ce graphe admet un arbre couvrant (en rouge):



Arbres et arbres couvrants

- Théorème
  - Soit $G = (V , E)$ un graphe non orienté.
  - Si G est connexe alors G possède un arbre couvrant.
- Définition
  - Soit $G = (V , E)$ un graphe orienté ou non.
  - On dit que G est valué si l'on attribue à chacun de ses arcs (ou arêtes) une valeur numérique.
  - Un graphe est dit à valuations positives si toutes ces valeurs sont positives ou nulles.
  - Dire que G est valué revient à dire qu'il existe une application v appelée valuation, définie sur E et à valeurs réelles :

$$\begin{array}{rccc} V : & E & \rightarrow & R \\ & (x, y) & \rightarrow & v(x, y) \end{array}$$

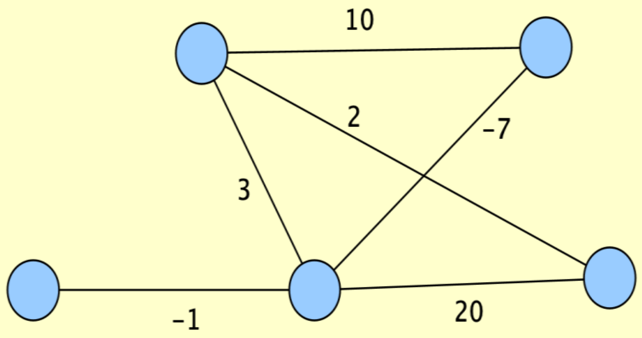
Arbres et arbres couvrants

- À préciser que la définition concerne un graphe valué aux arcs (arêtes)
- On peut attribuer une valeur aux sommets : graphe valué aux sommets.
- Si le graphe est non orienté, l'application valuation est symétrique.

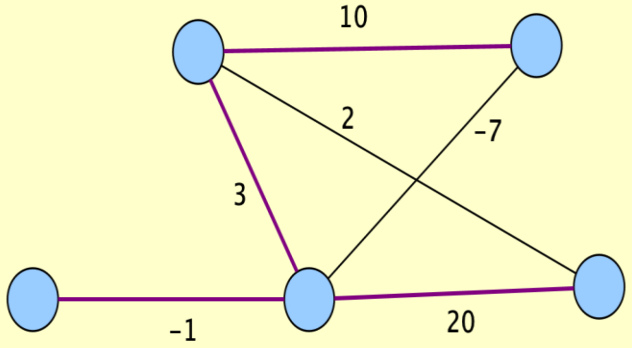
- Définition :
  - Soit $G = (V , E)$ un graphe non orienté à valuations quelconques.
  - Un **arbre couvrant de poids minimal** de G est un arbre couvrant de G dont la valuation est la plus petite parmi celles de tous les arbres couvrants de G .

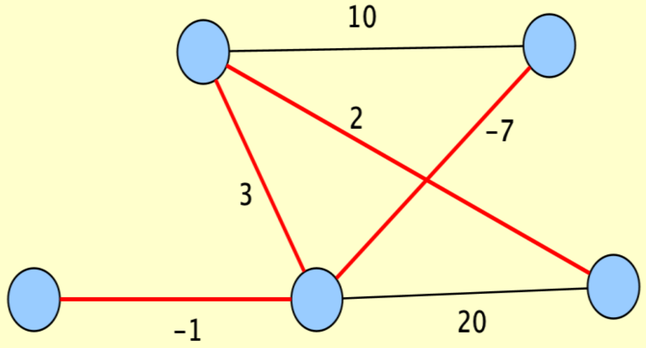
Arbres et arbres couvrants

- Théorème:
  - Soit G = (V , E) un graphe non orienté à valuations quelconques.
  - Si G est connexe alors G possède un arbre couvrant de poids minimal.
- Exemple : Considérons le graphe G non orienté valué



- Ce graphe admet un arbre couvrant :
  - de poids non minimal (mauve)
  - un autre de poids minimal (rouge)





Arbres et arbres couvrants

- Arbre couvrant de poids minimal : Applications
  - Minimisation du coût de construction d'un réseau de communication, connaissant le coût de chaque liaison possible entre les nœuds du réseau.
  - Objectifs :
    - Tous les nœuds du réseau puissent communiquer entre eux

    -> Rechercher un sous-graphe couvrant connexe
    - Établir un minimum de liaisons

    -> Rechercher un arbre couvrant
    - Coût global soit le plus faible possible

    -> rechercher un arbre couvrant de poids minimal.

Algorithme de Kruskal

- Algorithme de recherche d'un arbre couvrant de poids minimal dans un graphe non orienté connexe à valuations quelconques.
- Si n est l'ordre du graphe, l'algorithme procède en au moins n – 1 itérations.
- Démarrer à partir d'un sous-graphe contenant initialement les n sommets du graphe mais ne possédant aucune arête,
- Rajouter des arêtes une par une :
    - Au début de chaque itération, chercher l'arête de plus petite valuation parmi celles non encore traitées.
    - Deux cas de figures:
        - Si l'ajout de cette arête dans le sous-graphe en construction crée un cycle simple :

          -> Passer à l'arête suivante,
        - sinon on l'ajoute.
    - Dans les deux cas l'arête choisie est considérée comme traitée
    - Si card(arêtes) = n-1 arrêter sinon Ré-itérer.

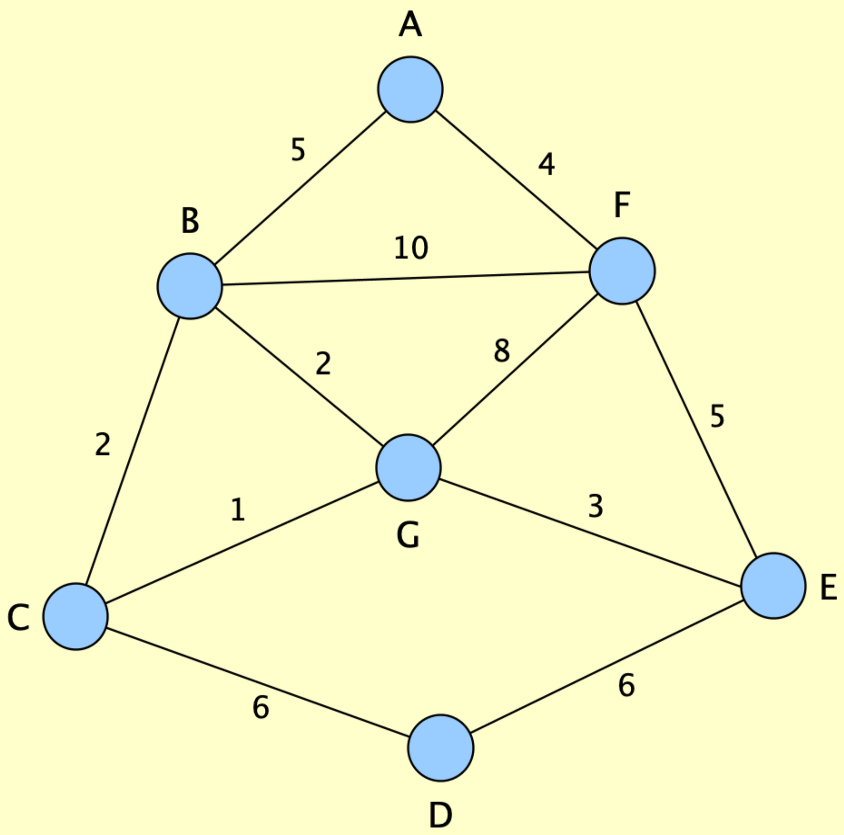
Algorithme de Kruskal

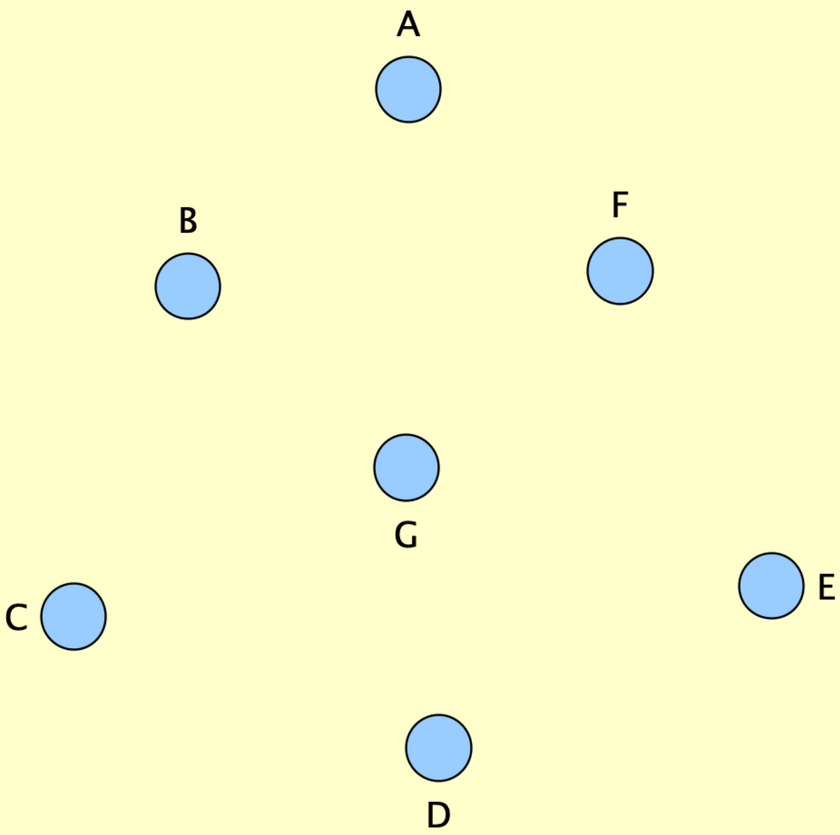
- Algorithme de Kruskal
  - Soit $G = (V , E)$ un graphe non orienté, connexe et à valuations quelconques d'ordre n.
  - Soit T l'ensemble des arêtes qui constitueront l'arbre cherché à la fin de l'algorithme.
  - Soit F l'ensemble des arêtes non encore traitées.

Initialisation	**Traitement**
$T \leftarrow \emptyset$	TantQue $card\ (T) < n - 1$
$F \leftarrow E$	Choisir une arête e $\in F$ de valuation minimale
	$F \leftarrow F \setminus \{e\}$
	Si $T \cup \{e\}$ est sans cycle
	$T \leftarrow T \cup \{e\}$
	Finsi
	FinTantQue

Algorithme de Kruskal

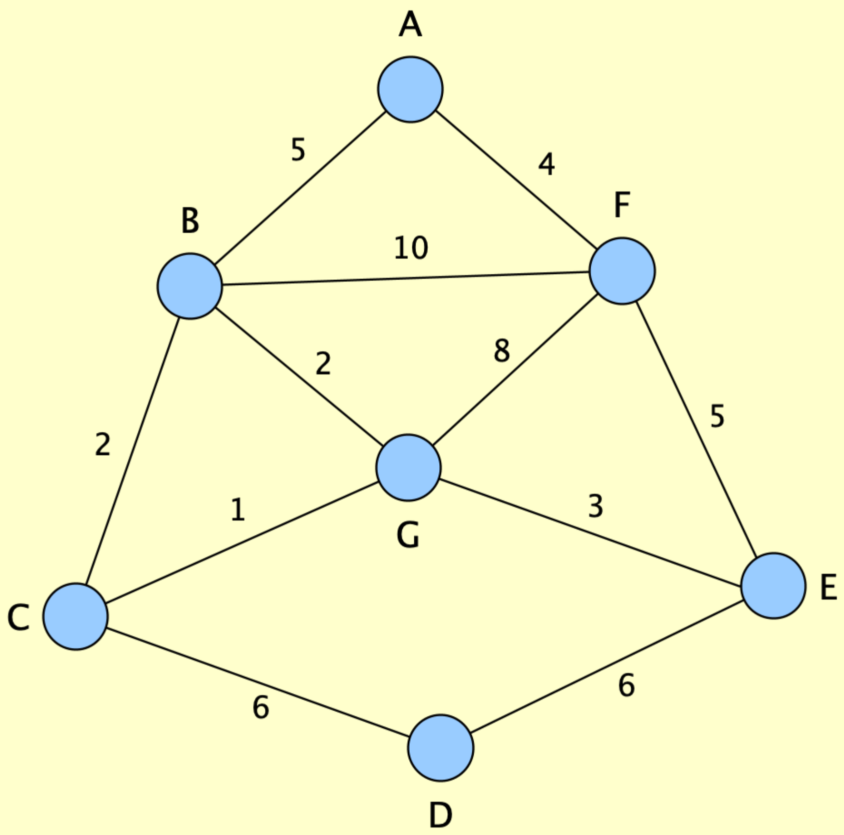
- Exemple:

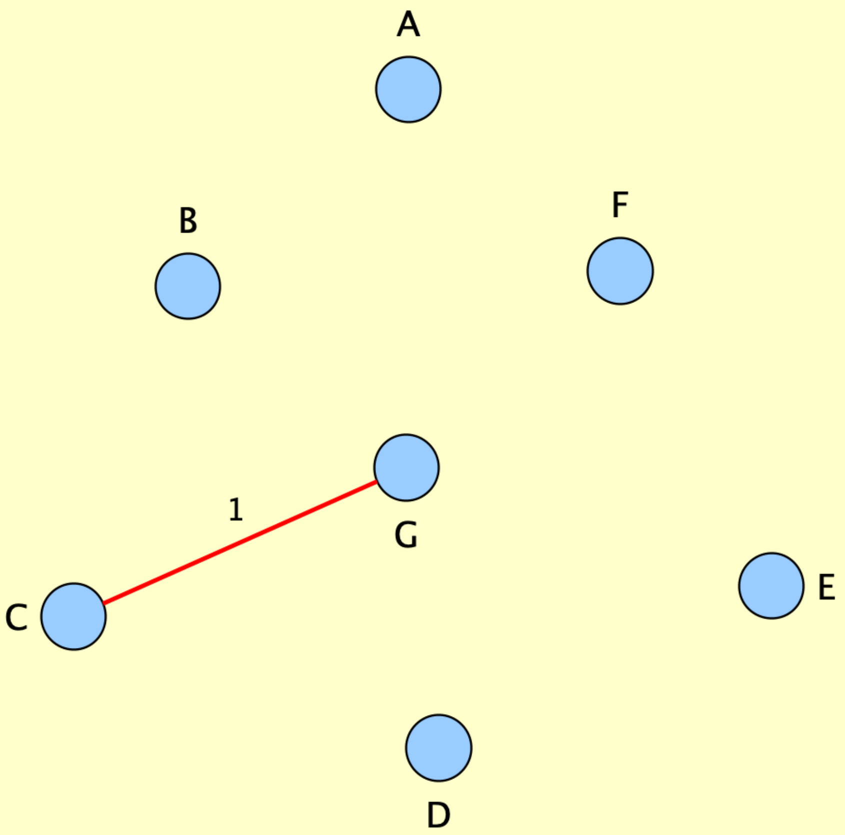




Algorithme de Kruskal

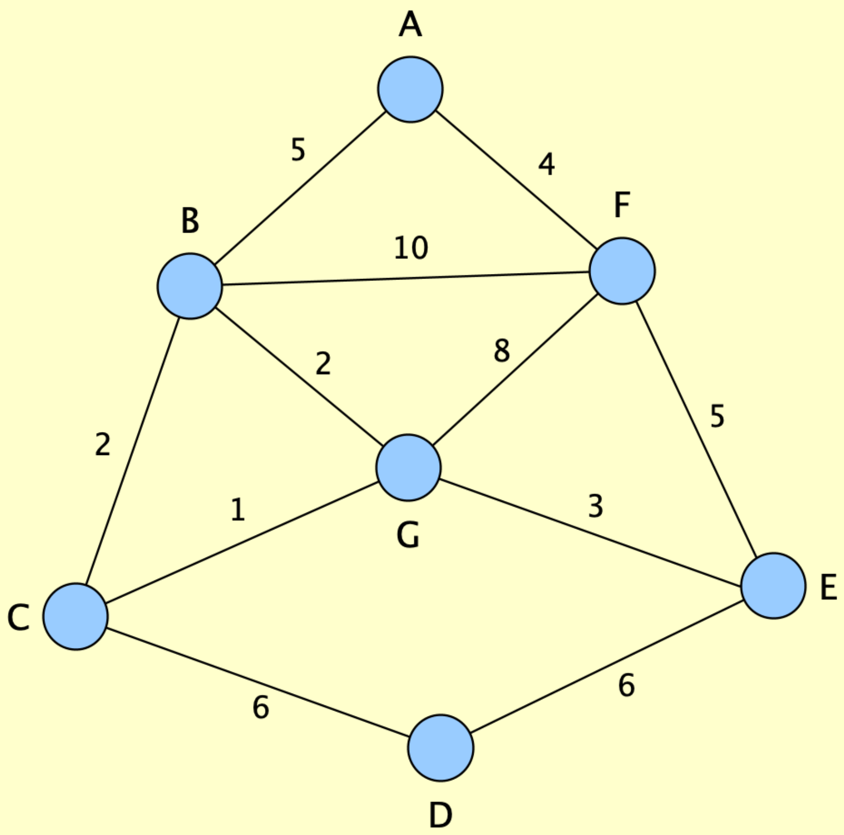
- Exemple:

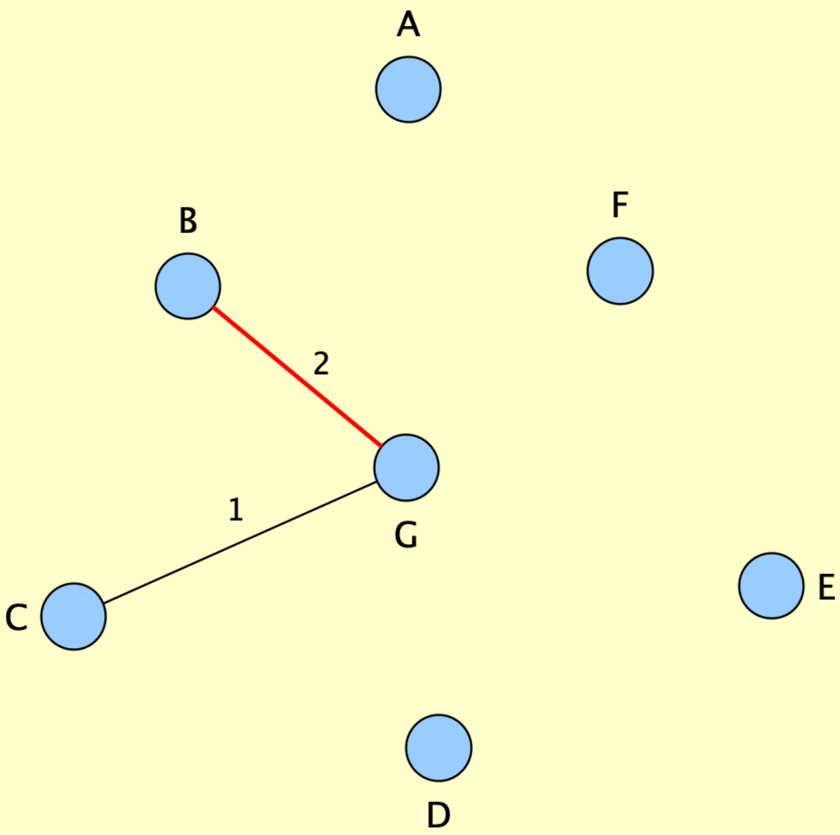




Algorithme de Kruskal

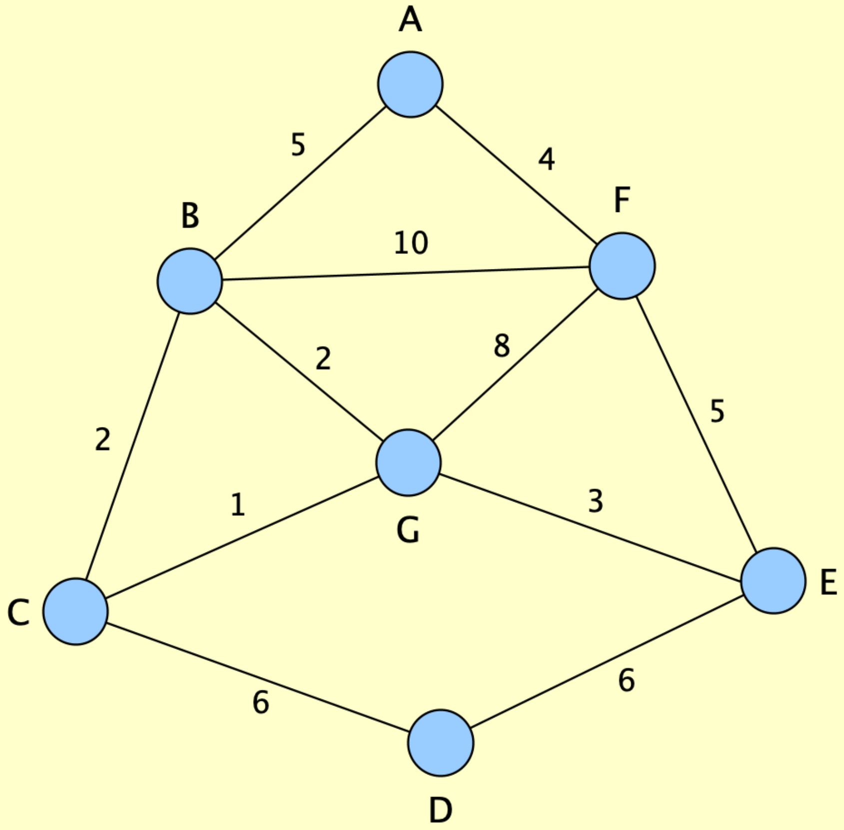
- Exemple:

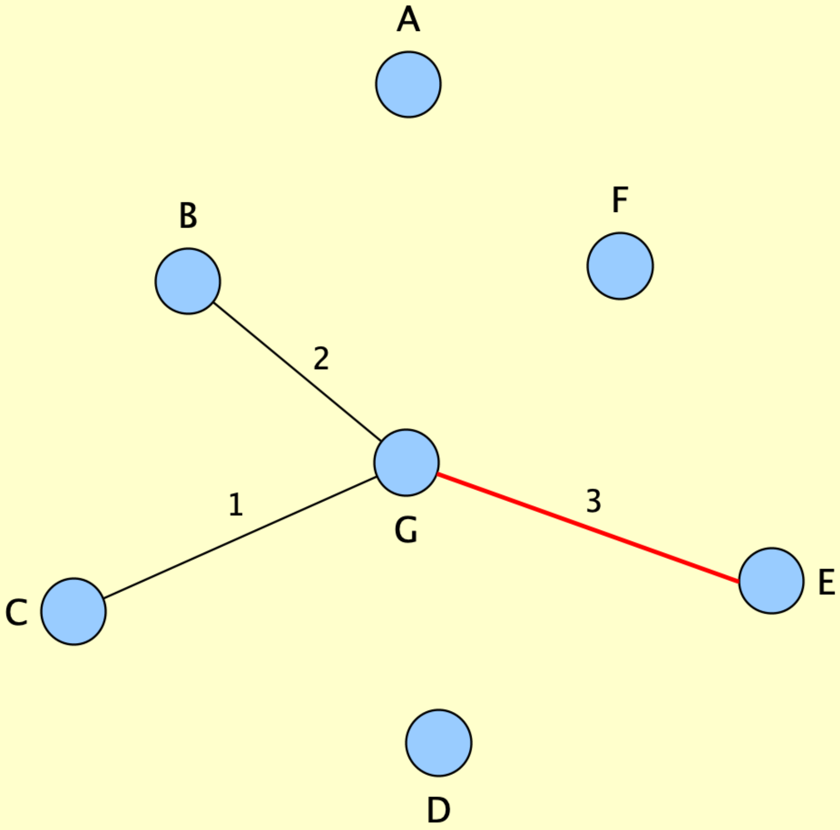




Algorithme de Kruskal

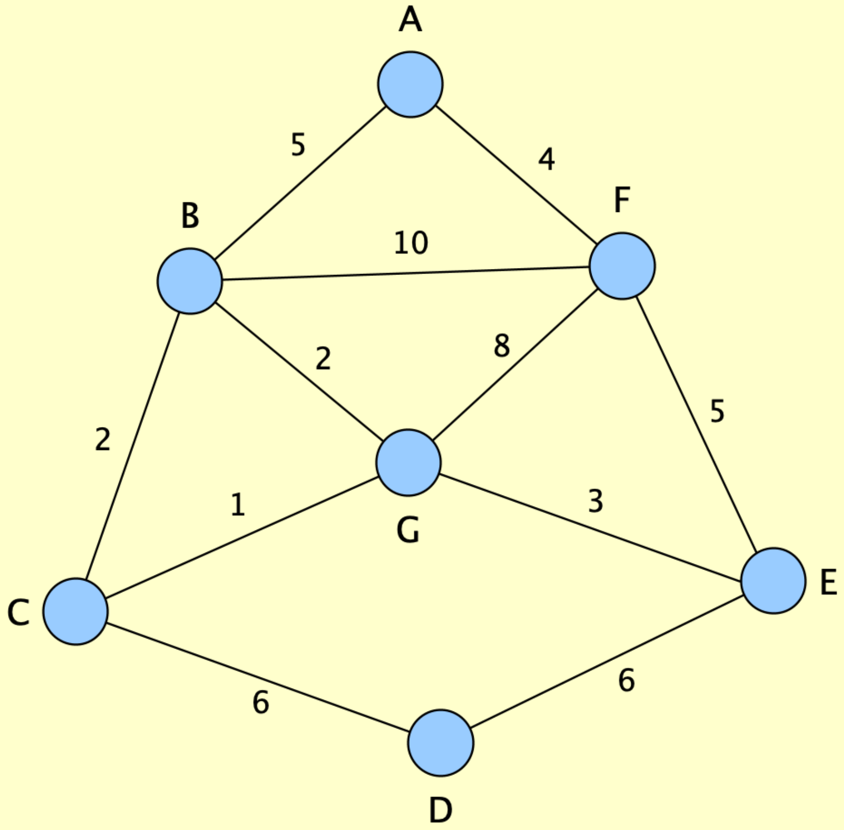
- Exemple:

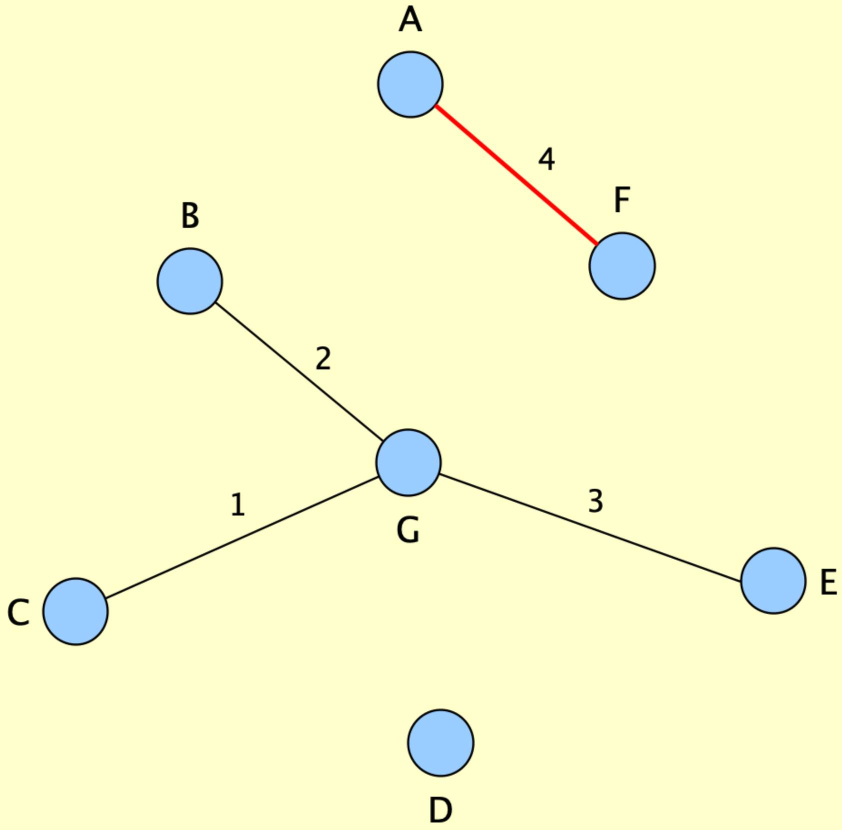




Algorithme de Kruskal

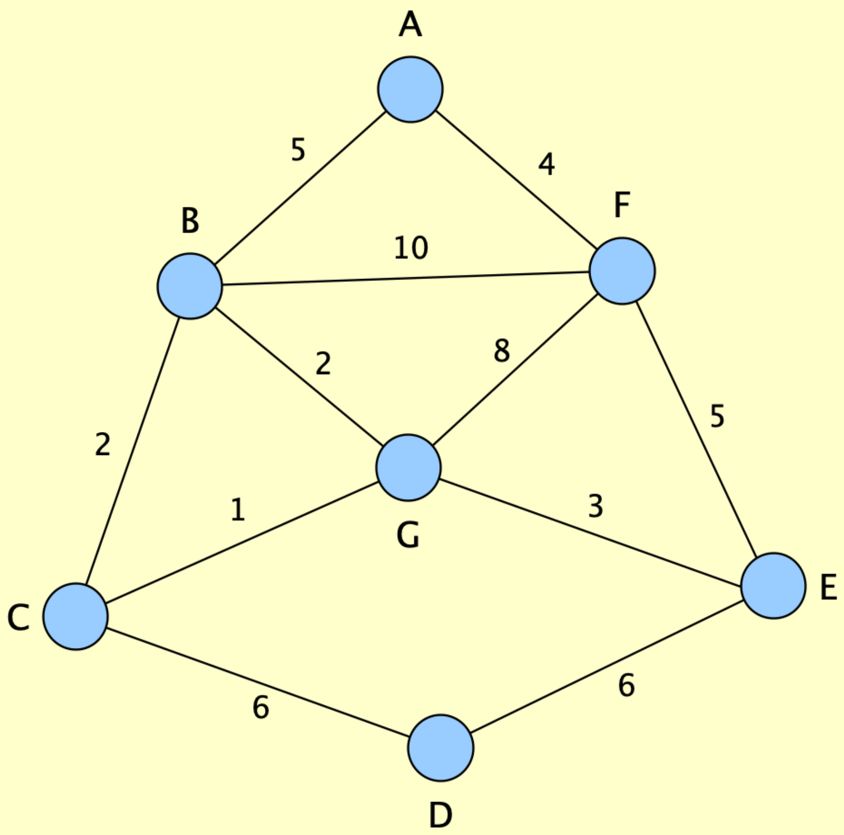
- Exemple:

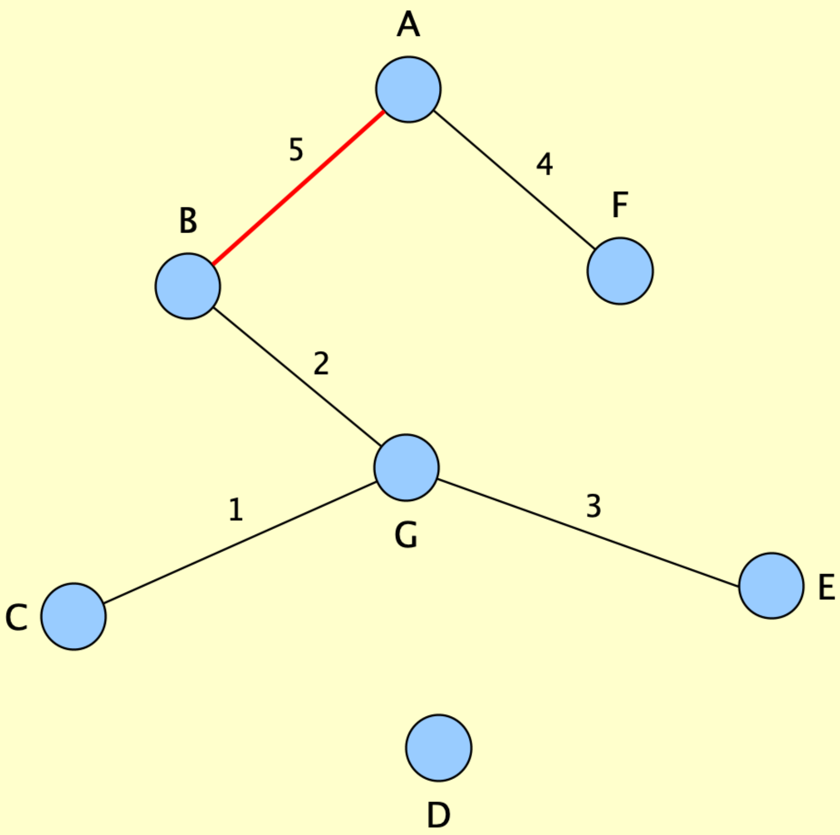




Algorithme de Kruskal

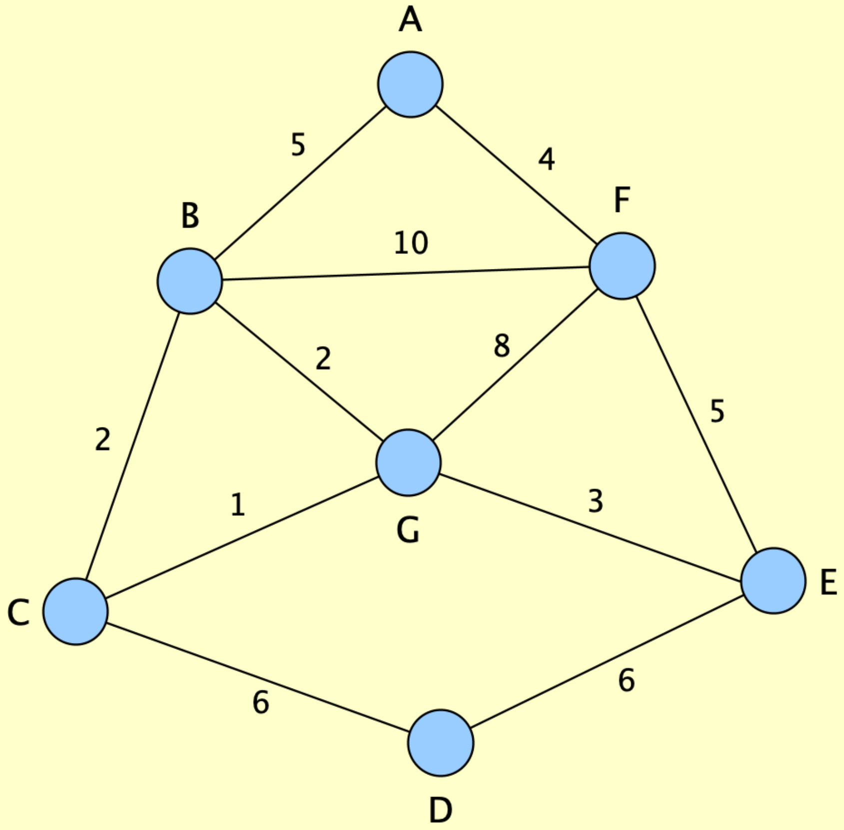
- Exemple:

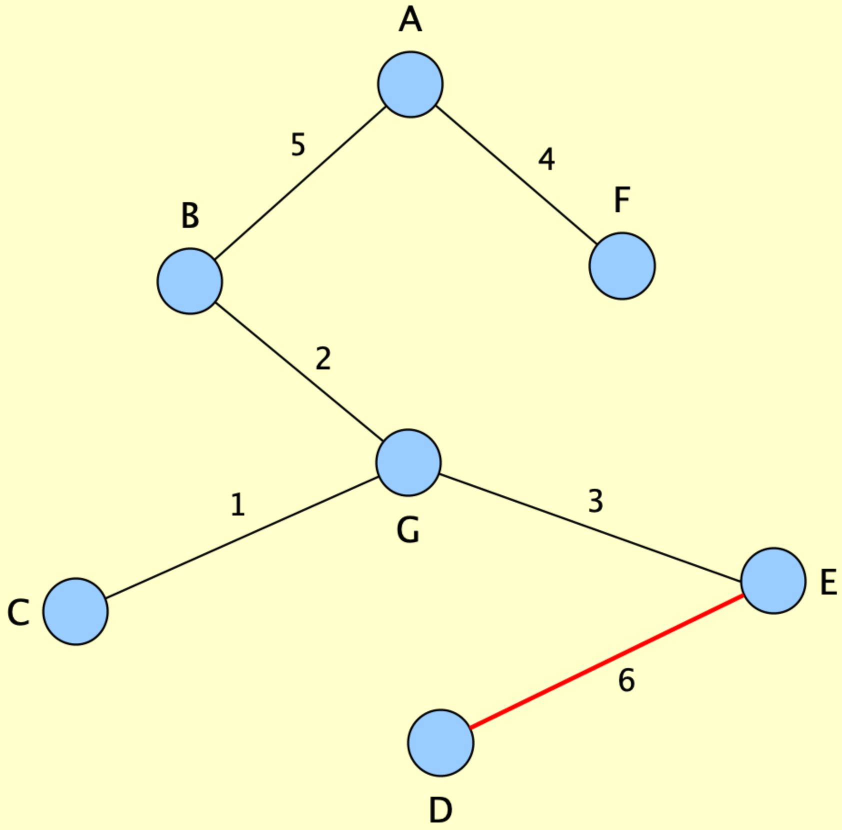




Algorithme de Kruskal

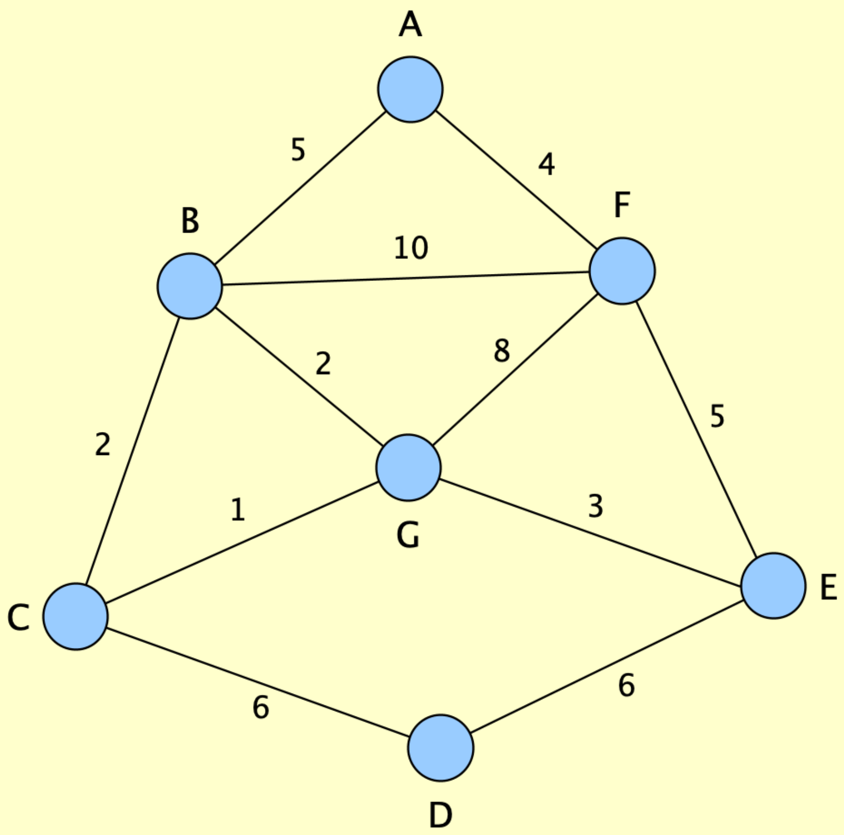
- Exemple:

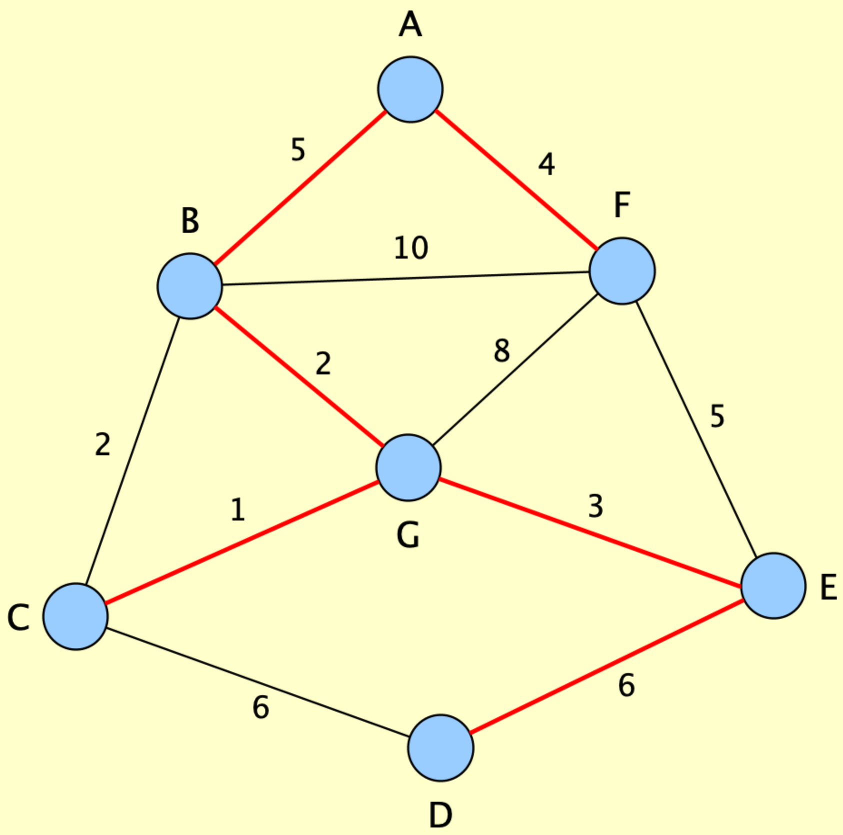




Algorithme de Kruskal

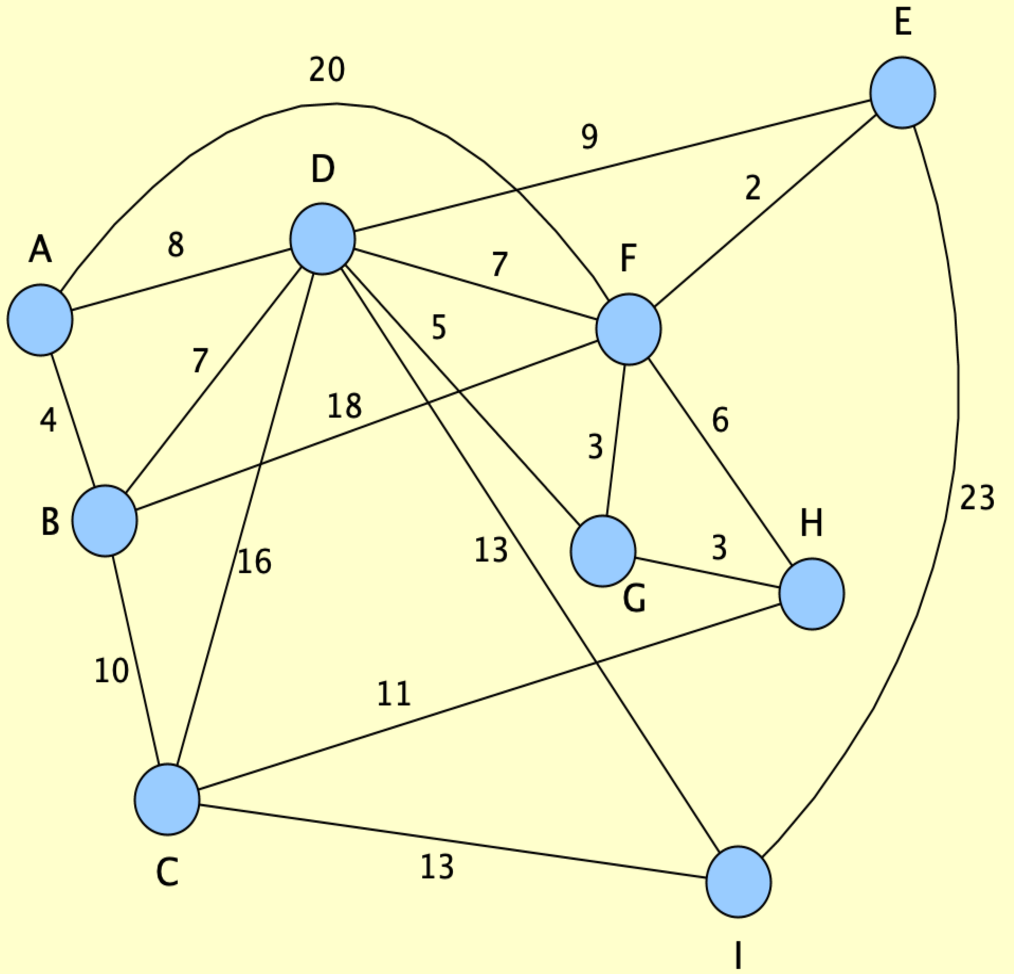
- Exemple:





Algorithme de Kruskal

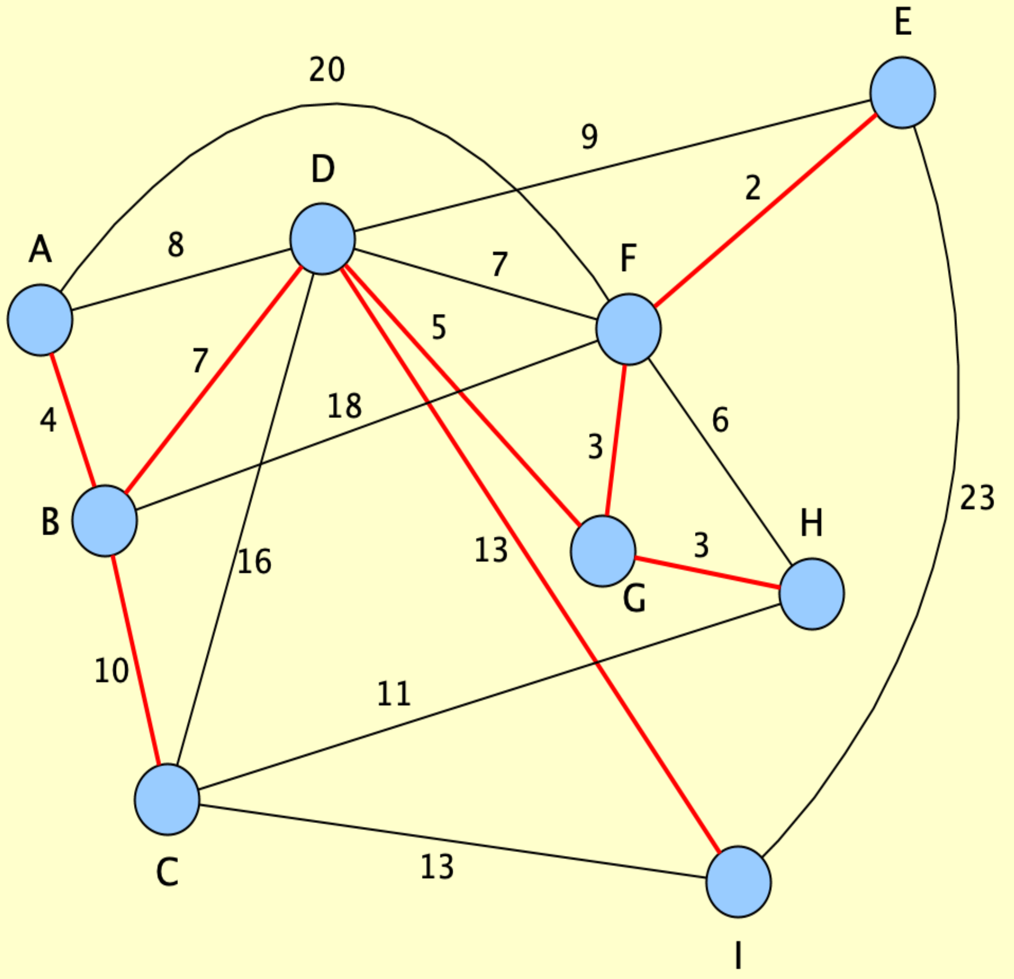
- Exercice :
  - Considérons le graphe non orienté valué ci-contre :



  - Appliquer l'algorithme de Kruskal à ce graphe,
  - en indiquant l'ordre dans lequel les arêtes ont été ajoutées.

Algorithme de Kruskal

- Correction : On obtient



  - Ordre d'insertion des arêtes :

{E,F}, {F,G}, {G,H}, {A,B}, {D,G}, {B,D}, {B,C} et {D,I}.

Algorithme de Prim

- Algorithme de recherche d'un arbre couvrant de poids minimal dans un graphe non orienté connexe à valuations quelconques.
- Si n est l'ordre du graphe, l'algorithme procède en n - 1 itérations.
- Démarrer à partir d'un sous-graphe ne contenant initialement qu'un seul sommet, choisi arbitrairement,
- Construire l'arbre recherché à partir de ce sommet, en ajoutant au fur et à mesure des sommets et arêtes :
  - Au début de chaque itération, ajouter à l'arbre en construction l'arête de plus petite valuation reliant un sommet de l'arbre à un sommet n'y appartenant pas encore.
  - On ajoute ce sommet à l'arbre.
  - Ré-itérer
- L'algorithme prend fin dès que le sous-graphe possède n - 1 arêtes (n sommets).

Algorithme de Prim

- Soit $G = (V , E)$ un graphe non orienté, connexe et à valuations quelconques d'ordre n.
- Soit T l'ensemble des arêtes qui constitueront l'arbre cherché à la fin de l'algorithme.
- Soit S l'ensemble des sommets constitué des extrémités des arêtes de T.
- Soit x_0 le sommet arbitrairement choisi à partir duquel on va construire l'arbre.

Initialisation

$T \leftarrow \emptyset$
$S \leftarrow \{x_0\}$

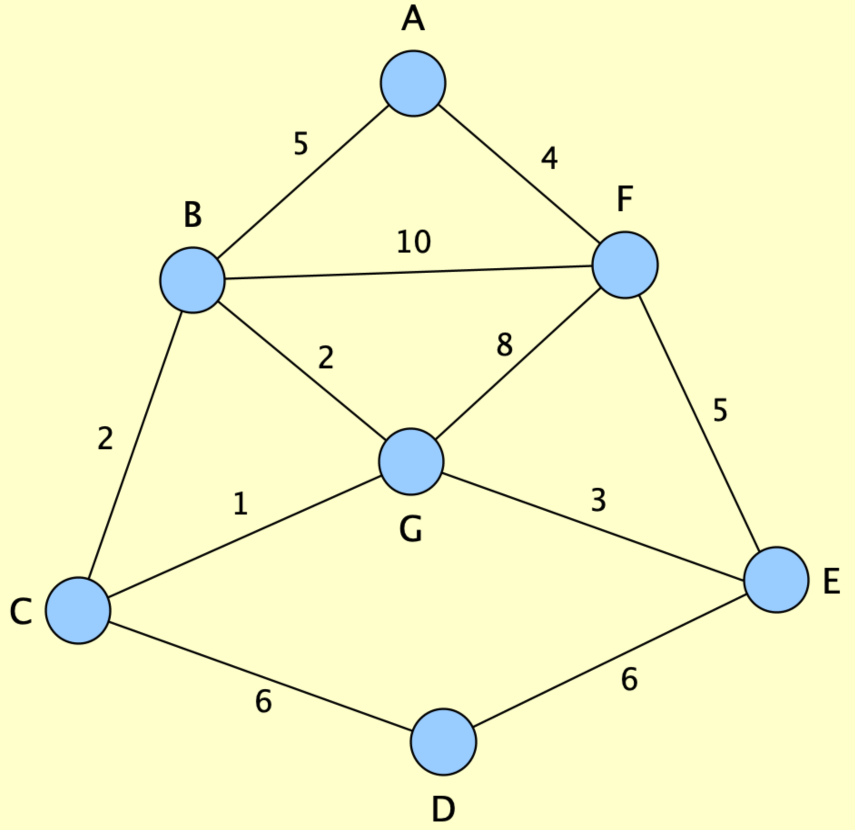
Traitement

TantQue $card\ (T\ ) < n - 1$
  Choisir une arête $e = \{x, y\}$ de valuation minimale
  telle que $x \in S$ et $y \in V \setminus S$

  $T \leftarrow T \cup \{e\}$ est sans cycle
  $S \leftarrow S \cup \{y\}$
FinTantQue

Algorithme de Prim

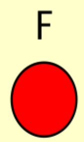
- Exemple
  - Considérons le graphe G orienté valué connexe ci-contre :



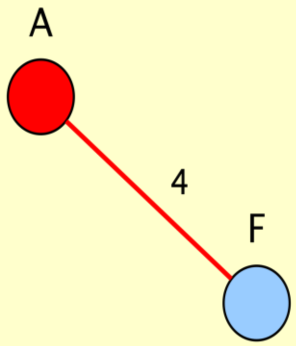
  - Choisissons arbitrairement le sommet F comme premier sommet de notre arbre en construction

Algorithme de Prim

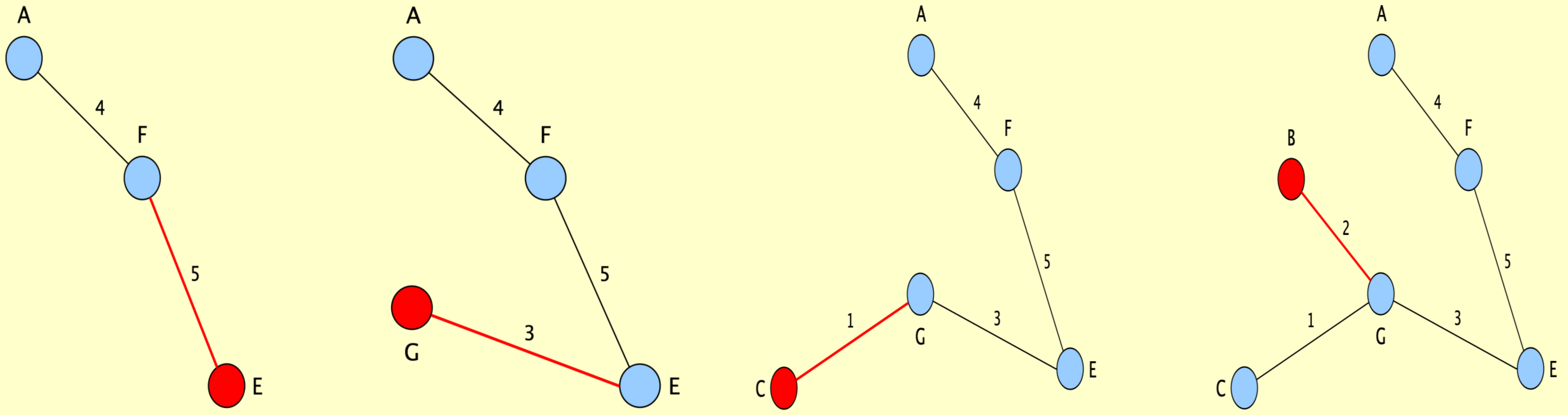
- Choisissons arbitrairement le sommet F comme premier sommet de notre arbre en construction :



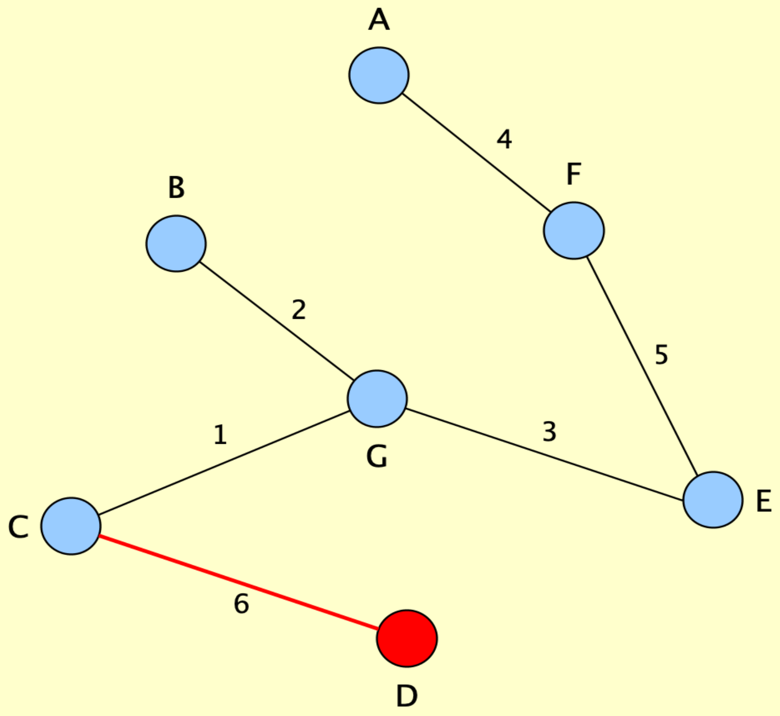
- On ajoute ensuite les sommets et arêtes au fur et à mesure, en faisant figurer le dernier ajout en rouge dans les graphes suivants.

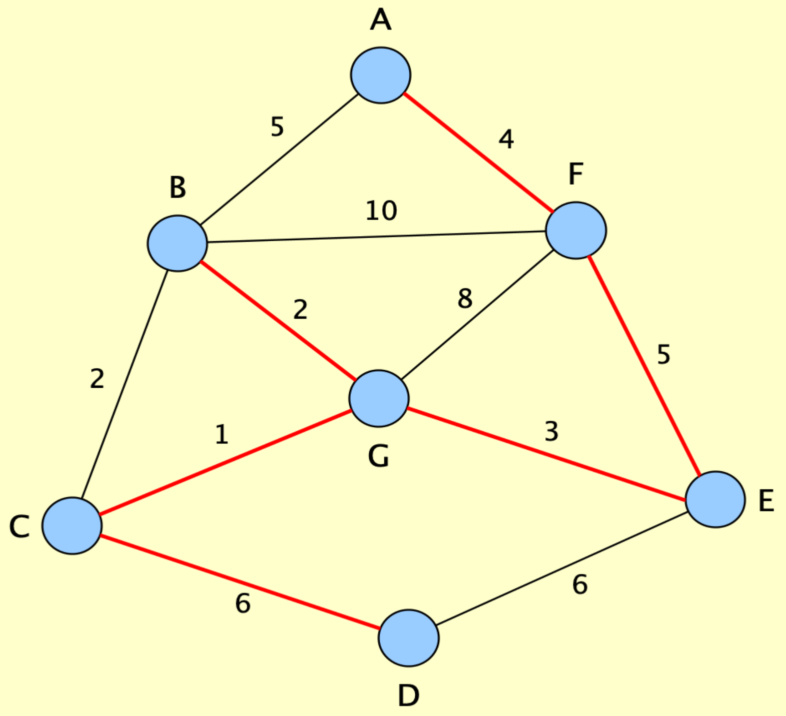


Algorithme de Prim



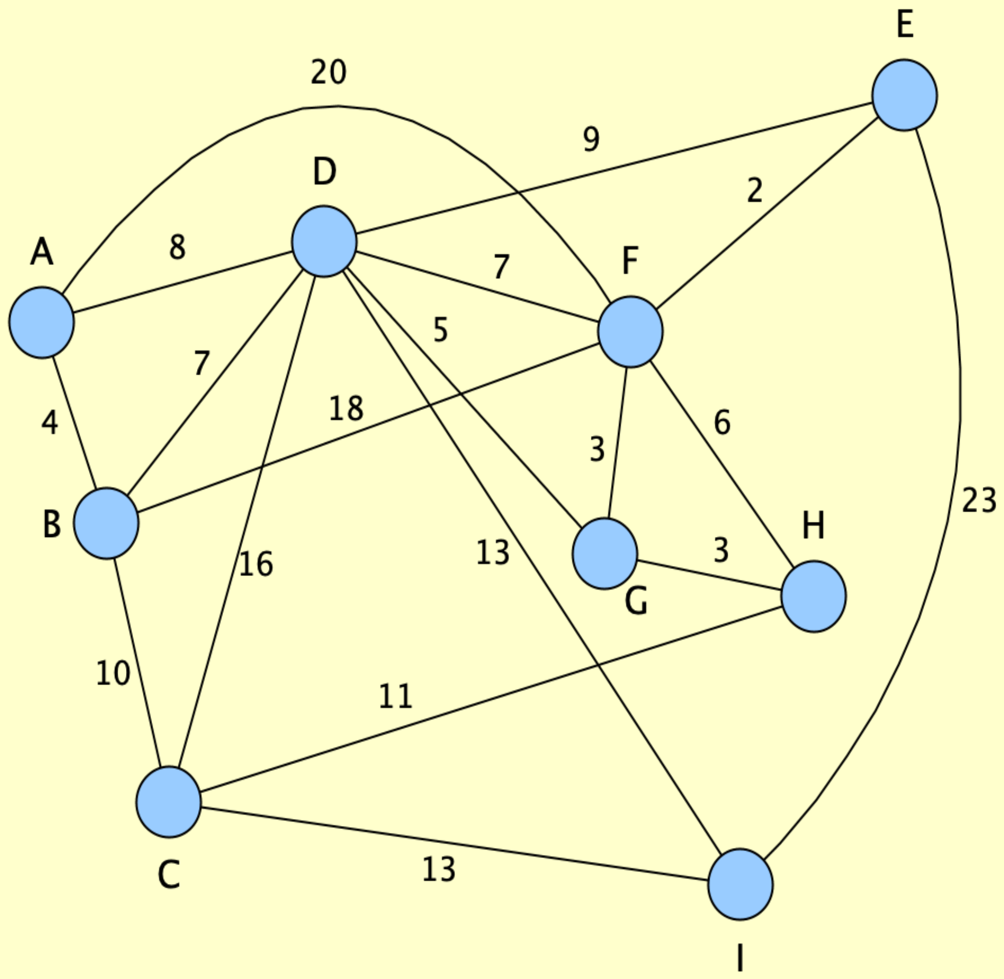
Algorithme de Prim





Algorithme de Prim

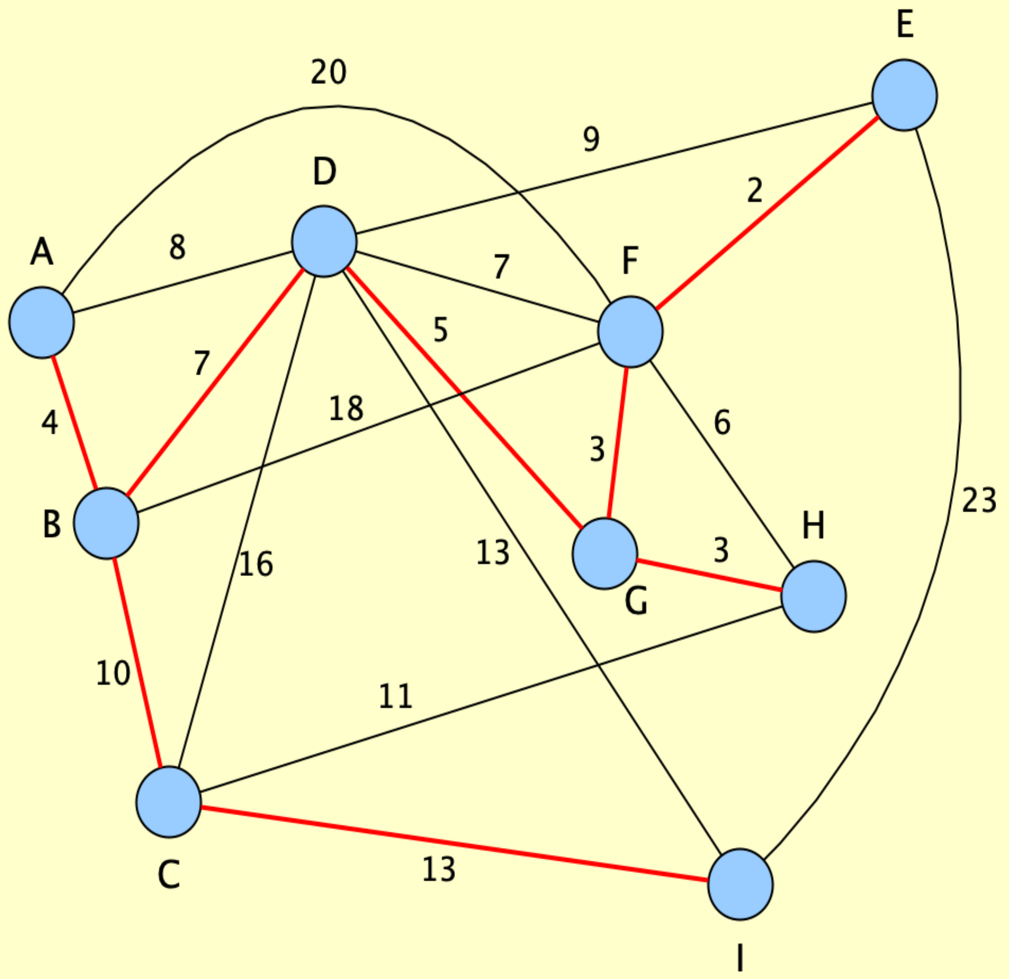
- Exercice :
- Considérons le graphe non orienté valué suivant :



- Appliquer l'algorithme de Prim à ce graphe. Indiquer en particulier l'ordre dans lequel les arêtes ont été ajoutées.

Algorithme de Prim

- Correction : On obtient



- Ordre d'insertion des arêtes :
  {B,C}, {A,B}, {B,D}, {D,G}, {F,G}, {E,F}, {G,H} et {C,I}.