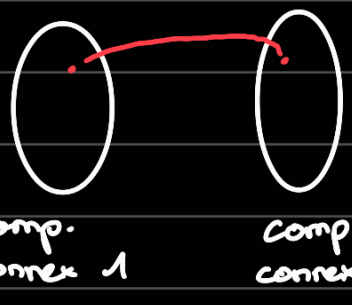
CM 12/11

Arbre couvrant de poids minimum ?

(Rappel :) Parmi les arêtes non choisies.
On prend celle(s) de poids minimum.
et on la marque uniquement si elle ne crée pas un cycle avec l'arbre couvrant.

Astuce pour détecter des cycles.

Soit G :



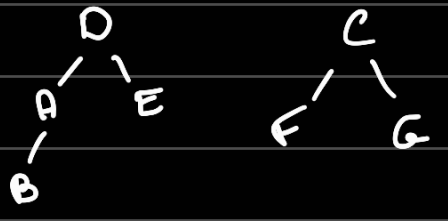
En ajoutant l'arête rouge, on est sûr de ne pas obtenir de cycle.

→ Nouvelle structure de donnée.

"Union-Find"

Idée: Stocker un ensemble S comme une arborescence.

Exemple: $\{A, B, D, E\}, \{C, F, G\}$:



(l'ordre des sommets n'importe pas)

- Sommets de S = noeuds
- $\forall x \in S$ on associe $\pi(x)$ son pointeur. ainsi on stocke la racine et en suivant le chemin $(x, \pi(x), \pi(\pi(x)), \dots)$

- On note $\text{rank}(x)$ le nbr d'étages après x.

Fonctions :

makeset(x) $O(1)$

$\pi(x) \leftarrow x$

$\text{rank}(x) \leftarrow 0$

Find(x)

while $x \neq \pi(x)$

$x \leftarrow \pi(x)$

return x.

Exemple : (cf poly page 8)

(A^1) veut dire $\text{rank}(A) = 1$

Pour fusionner 2 arbres :

$\pi(r_A) = r_B$ (on ajoute l'arbre A comme enfant de B)

(on peut évidemment aussi faire $\pi(r_B) = r_A$)

Si $h(A) > h(B)$ et $\pi(r_A) = r_B$

alors la hauteur finale augmente de 1.

Sinon (si $h(A) < h(B)$ et $\pi(r_A) = r_B$
alors la hauteur n'augmente pas
Ls et on cherche toujours à faire ça.

```
union(S, T):
    r_x ← find(x)
    r_y ← find(y)
    si r_x = r_y
        return
    si rank(r_x) > rank(r_y)
        π(r_y) ← r_x   (pour éviter d'augmenter la hauteur)
    sinon
        π(r_x) ← r_y
        si rank(r_x) = rank(r_y)
            rank(r_y) ++
```

Lemme : Soit x un sommet d'une arborescence A.
Si $rank(x) = k$
alors la s/s arborescence avec racine x
a au moins 2^k sommets.

Complexité de kruskal :

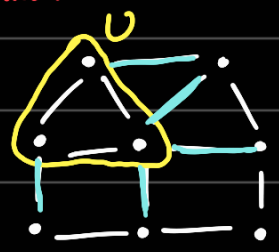
- makeset appelé n fois.
- Tri en $m\log(m)$ (car s/ partie des arêtes)
- find et union appelés m fois

$\log(n)$

donc kruskal : $O(m\log(m))$

Relation arbre-couvrant ↔ coupe de poids min

Rappel : Soit G :



La coupe est l'ensembles $\bar{U}$

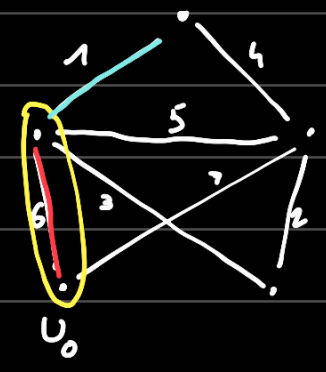
$$C = \{ \forall u \in U, \forall v \in V(G) \setminus U \mid uv \in E(G) \}$$

(ensemble de uv avec un unique sommet dans U

/ est la coupe sur U dans G

Algorithme de Prim :

Exemple pour comprendre :



1. - Choix arête de poids min
2. - Coupe de l'ens des arêtes choisis
3. - Choix de l'arête min de la coupe.

Entrées : $G = (V, E)$ pondéré

Sortie : Arbre couvrant de poids min.

- Pseudo-code (cf poly)
- Implémentation avec files de priorités (cf poly page 22)