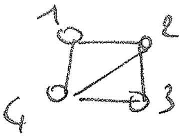
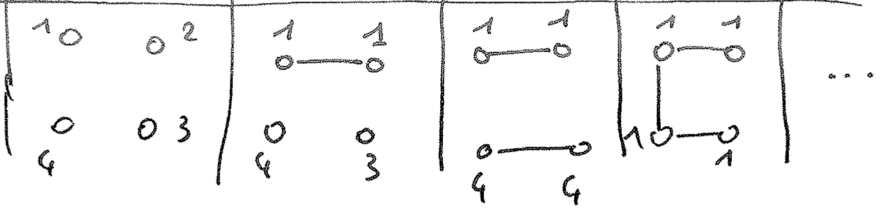
- Principe : . au début coup(x) = x pour tout x
  . On lit les arêtes, à chaque arête xy on fusionne coup(x) et coup(y).
- Ex :



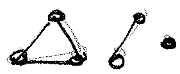
Arêtes lues	Avant	12	34	14	...
Fonction coup					...



- COMPOSANTE (G = (V, E))

1 [. pour tout $x \in V$ coup(x) ← x ;
2 . Pour tout $xy \in E$:
3   [. Si coup(x) ≠ coup(y) alors
4   [. aux ← coup(x)
5   . Pour tout $z \in V$ faire
6   [. Si comp(z) = aux alors coup(z) ← coup(y)
7 . Retourner coup.

Cas 1 → Réécrire COMPO. et Exemple non connexe :



- Analyse de l'algo :

① Terminaison : que des boucles "for" et des tests → OK.

② Correction : montrons que "coup(x) = coup(y) ⟺ ∃ xy-marche".
Par induction sur m (= |E(G)|)
- Si m = 0 c'est vrai
- Sinon, soit xy la dernière arête traitée L.2. Par induction la propriété était vraie avant le traitement de xy.
  . Cas 1 : Si coup(x) = coup(y), x et y sont déjà reliés par une xy-marche, rajouter xy ne change

rien aux composantes connexes

- cas 2 : $comp(x) \neq comp(y)$. A la fin de la ligne 6 tous les sommets précédemment marqués $comp(x)$ sont maintenant marqué comme $comp(y)$: les composantes de x et y ont bien fusionnées.

③ Complexité : on calcule le temps d'exécution (dans le pire des cas)

$L_1 \to O(n)$ $\quad$ $L_4 \to O(n)$ (le test L_3 est vrai au + $n-1$ fois, à chq fois, on fusionne 2 comp).

$L_2 \to O(m)$ $\quad$ $L_5 \to O(n^2)$

$L_3 \to O(m)$ $\quad$ $L_6 \to O(n^2)$

En tout $O(n^2+m) = O(n^2)$ (rappel : $m \leq \frac{n(n-1)}{2}$)

On verra + rapide ...

---

o $\boxed{\text{Pb du jour n° 1}}$: on va chercher des chemins ou marches ayant diverses propriétés. Regardons la propriété "passer par tous les sommets".

* Pbl. Existe-il une marche qui passe par tous les sommets ?

- Réponse : Oui ssi le graphe est connexe ⇝ on sait faire.

* Pbl (CHEMIN HAMILTONIEN) : Existe-il un chemin qui passe par tous les sommets ?

- C'est moins évident (regarder $K_{2,4}$ pour s'en convaincre).

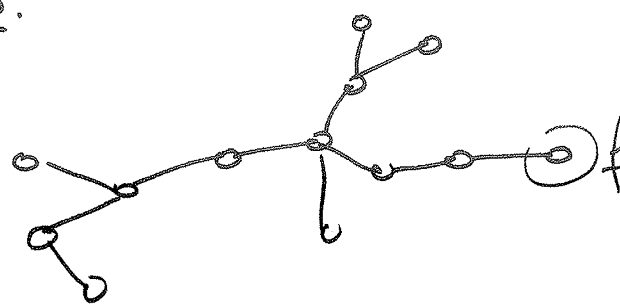
On n'a pas de bon algo pour calculer ça (le seul, à peu de chose près, est de tester toutes les permutations). En tout cas, on ne sait pas si il existe un algo polynomial pour ce problème, il est NP-complet (NP ne veut pas dire Non Polynomial cf → M1).

Clay Math Institute Il y a 1 M.$ pour un algo polynomial ou la preuve de son inexistence

Chap II : Arbres couvrants

I) Arbres.

· Ex :



· Un arbre $T=(V,A)$ est un graphe connexe et sans cycle.

Une feuille de T est un sommet de degré 1. Un arbre a-t-il toujours une feuille ? Oui :

· **Proposition** : tout arbre ayant au moins 2 sommets a au moins 2 feuilles.

Pr : (typique) Soit $v_1, \dots, v_t$ un chemin maximal (inextensible).

. v_1 n'a pas d'autre voisin que v_2, sinon on peut étendre le chemin ou créer un cycle → v_1 est une feuille.

. Idem pour v_t. ∎

· Enfin, remarquons que si f est une feuille de T, $T \setminus \{f\}$ (T privé de f) est encore un arbre. On peut continuer :

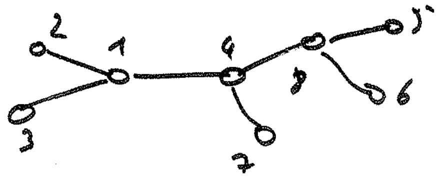
Soit T un arbre sur n élts :

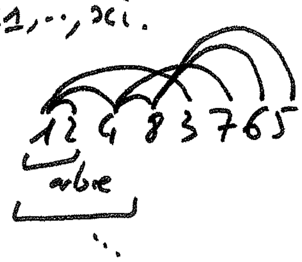
- On note f_n une feuille de T
- ——— f_{n-1} ——————— $T\setminus\{f_n\}$
- ——— f_{n-2} ——————— $T\setminus\{f_n, f_{n-1}\}$

$\vdots$

On obtient une énumération des sommets de T : $f_1, \dots, f_n$ appelé <u>schéma d'élimination</u> de T et qui vérifie : $\forall i$ x_i est une feuille du sous arbre de T réduit à $x_1, \dots, x_i$.

- <u>Ex</u> :



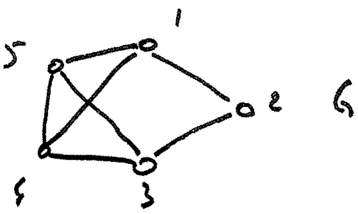


1 seule arête vers la gauche.

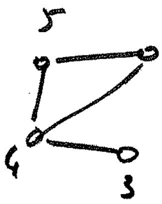
II) <u>Arbres couvrants</u>

- Soit $G = (V, E)$ un graphe. Un sous-graphe $H = (V', E')$ avec $V' \subseteq V$ et $E' \subset E$ est un <u>sous-graphe</u> de G. Si de plus $V' = V$ alors H est un <u>sous graphe couvrant</u> de G.

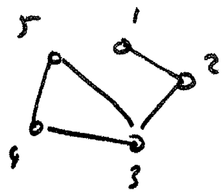
<u>Ex</u> :



G



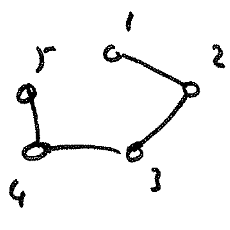
sous graphe de G.

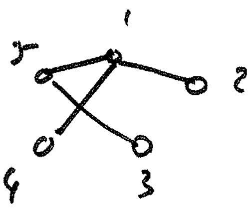


couvrant

- Si H est un ssgraphe couvrant de G et H est un arbre, on dit que H est un <u>arbre couvrant</u> (!) de G.

- <u>Ex</u> : pour G ↑





il y a possiblement plusieurs ...

- <u>Théo</u> : Un graphe G est connexe ssi il admet un arbre couvrant.

<u>Pr</u> : $\Leftarrow$ immédiat xy-chemin de T $\Rightarrow$ xy-chemin de G.

$\Rightarrow$ Supposons G connexe

Si G n'a pas de cycle G est un arbre OK.

Sinon G a un cycle $x_0 x_1 x_2 \ldots x_k$, si on supprime l'arête $x_0 - x_1$, G reste connexe : toute marche empruntant l'arête $x_0 x_1$ peut être "réaiguillée" en utilisant le chemin $x_0 x_k \ldots x_1$ à la place.

On applique l'algo suivant :

```
Tant qu'il existe un cycle C dans G
    [ Enlever une arête de C.
```

L'algo retourne un arbre couvrant de G.

Rq : . On verra + rapide que cet algo-là pour le calcul d'un A.C.
. Un arbre couvrant est un certificat de connexité.

Cours 2.

III] Un algo de calcul d'un arbre couvrant.

ARBRECOUVRANT $(G=(V,E))$.

ENTREE : Graphe connexe $G=(V,E)$ donné par liste d'arêtes.
SORTIE : Un ens. $A \subseteq E$ d'arêtes t.q. (V,A) soit un arbre couvrant G.

```
A = ∅
Pour tout x ∈ V   c(x) ← x ;
Pour tout xy ∈ E
    [ Si c(x) ≠ c(y) alors
        [ • aux ← c(x) ;                    NEW
          • A ← A ∪ {xy}
          • Pour tout z ∈ V faire
              [ • Si c(z) = aux alors c(z) ← c(y) ;
```