Définition : Un tri topologique d'un graphe est un ordre total < pour tout arc uv ∈ E tq $u < v$ (les arcs peuvent être placés sur une droite et tous les arcs vers la m̂ direction)

Théorème : Un graph G admet un tri topologique $\Leftrightarrow$ G est acyclique

Remarque : En effectuant un DFS avec les tableaux pre et post, on peut ensuite trier les sommets par nombre post décroissant

CONNEXITÉ DANS LES GRAPHES ORIENTÉS

- Plus compliqué que pour Graphes non-orientés.
- On dit que u et v sont connectés $\Leftrightarrow$ ∃ un chemin de u vers v et de v vers u.

Propriété 1: Si la procédure explorer(G, s) est lancée au sommet u, alors elle terminera précisément lorsque tous les sommets atteignables à partir de u auront été visités.

Donc, si u est un sommet d'une composition fortement connexe G_i de G qui est un "puits" dans le graphe contracté G', alors explorer va parcourir uniquement les sommets de G_i.

Propriété 2: Soient G_i et G_j des composants fortement connexes de G. S'il existe un arc d'un sommet de G_i vers un sommet de G_j, alors

$$\max\left(\{post(v) : \forall v \in V(G_i)\}\right) > \max\left(\{post(v) : \forall v \in V(G_j)\}\right)$$

Propriété 3: Le sommet avec la valeur maximale de post dans un parcours en profondeur appartient à une composante fortement connexe de type "source".

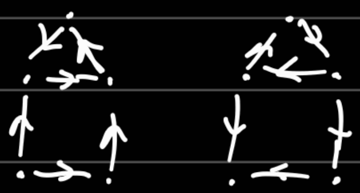
<u>Conséquence</u> : On peut trier des composantes G_i fortements connexes par ordre décroissant de leur nombre $\max(\{post(v) : v \in V(G_i)\})$.

Pourtant, on voudrait plutôt trouver un sommet dans un composant connexe de type puits.

Il suffit de considérer le graphe inverse

$$G^R = (V, E^R), \quad \text{où } \forall (u,v) \in E \Leftrightarrow (v,u) \in E^R$$

<u>Exemple:</u>



Algorithme de calcul des composantes fortement connexes.

1) Executer DFS s/ G^R
2) Executer DFS s/ G, dans l'ordre de nombre post décroissant trouvé dans 1).