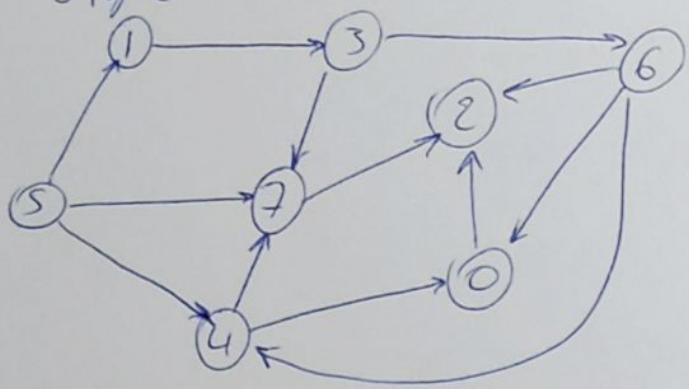
DEV : Tri topologique pour les graphes orientés acycliques

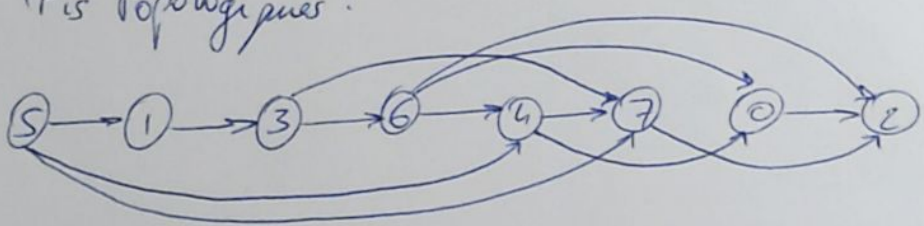
Inspiration : Informatique MP2I/MPI de Balbonski,... partie 8.3.1.3

Définition 1 : On appelle tri topologique d'un graphe orienté (V,E) une permutation de ses sommets telle que pour tout arc $(u,v) \in V$, u apparaît avant v dans la liste.

Exemple 2 : Graphe



Tris topologiques :



5, 1, 3, 6, 4, 0, 7, 2

Remarque 3 : Utilisation des graphes orientés acycliques dans la modélisation des problèmes d'ordonnancement : un arc $u \longrightarrow v$ indique alors que u doit être effectué avant v.
Exemple d'utilisation : Makefile pour déterminer l'ordre de compilation.

Propriété 4 : Un graphe orienté est acyclique si et seulement s'il possède un tri topologique de ses sommets.

Preuve. ⇐ Il n'y a aucun arc arrière dans le tri topologique, il ne peut donc pas y avoir de cycle.

⇒ On montre l'existence par l'algorithme suivant :

Algorithme 5 : Tri topologique
Entrée : $G = (V, E)$ graphe orienté acyclique
Sortie : Tri topologique de G

```
visites(t) ← Blanc pour tout t de V
ordre ← []
fonction dfs(s):
    Si visites(s) = Blanc, alors
        visites(s) ← Gris
        Pour tout successeur t des :
            dfs(t)
        visites(s) ← Noir
        Insérer s au début de ordre
    Sinon si visites(s) = Gris :
        Erreur "Le graphe est cyclique"
Pour tout sommet s de V :
    Si visites(s) = Blanc :
        dfs(s)
Renvoyer ordre
```

Propriété 6 : L'algorithme 5 termine et a une complexité temporelle de $O(|V| + |E|)$.

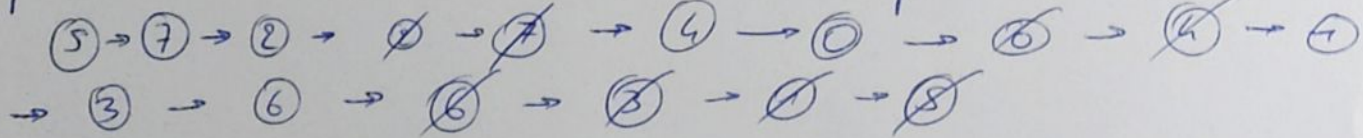
Preuve : Cet algorithme est exactement le parcours en profondeur récursif (à la juste ordre près) : il termine donc et a une complexité de $O(|V|+|E|)$.

Propriété 7 : L'algorithme 5 renvoie un tri topologique de G.

Preuve : Tous les sommets sont appelés dans dfs, il sont donc bien tous ajoutés à ordre. De plus, les couleurs assurent qu'ils ne le sont qu'une fois. Il suffit donc de montrer que pour tout $(u,v) \in E$, u est avant v dans ordre. Soit $(u,v) \in E$: on montre que dfs(u) termine après dfs(v). On considère le premier appel à dfs(u).

- si dfs(v) a déjà été fait et est terminé, v est déjà dans ordre et u est avant v (insertion en début de liste)
- si dfs(v) est déclenché par cet appel, alors dfs(v) finira avant dfs(u)
- sinon, dfs(u) est déclenché par un appel de dfs(v), donc il existe un chemin de v à u : G est cyclique : absurde.

Exemple 8 : En reprenant le schéma de l'exemple 2 :



↳ Tri topologique : 5, 1, 3, 6, 4, 0, 7, 2