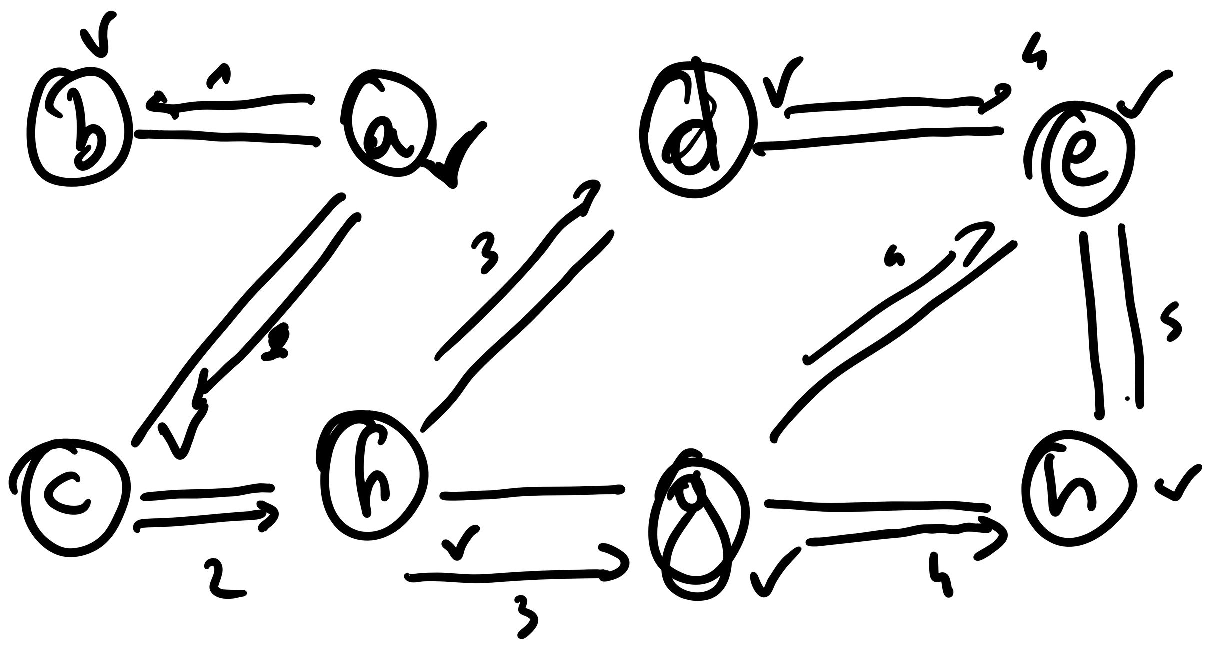
Parcours en profondeur (DFS)

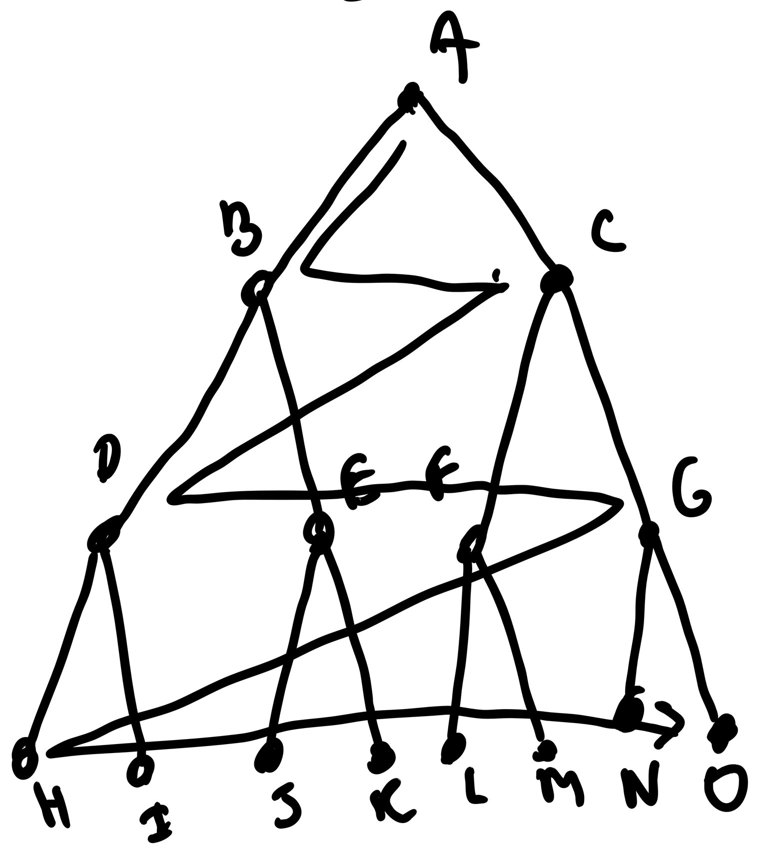
Entrée : Un graphe $G = (V, E)$ et un sommet $r \in V$

début

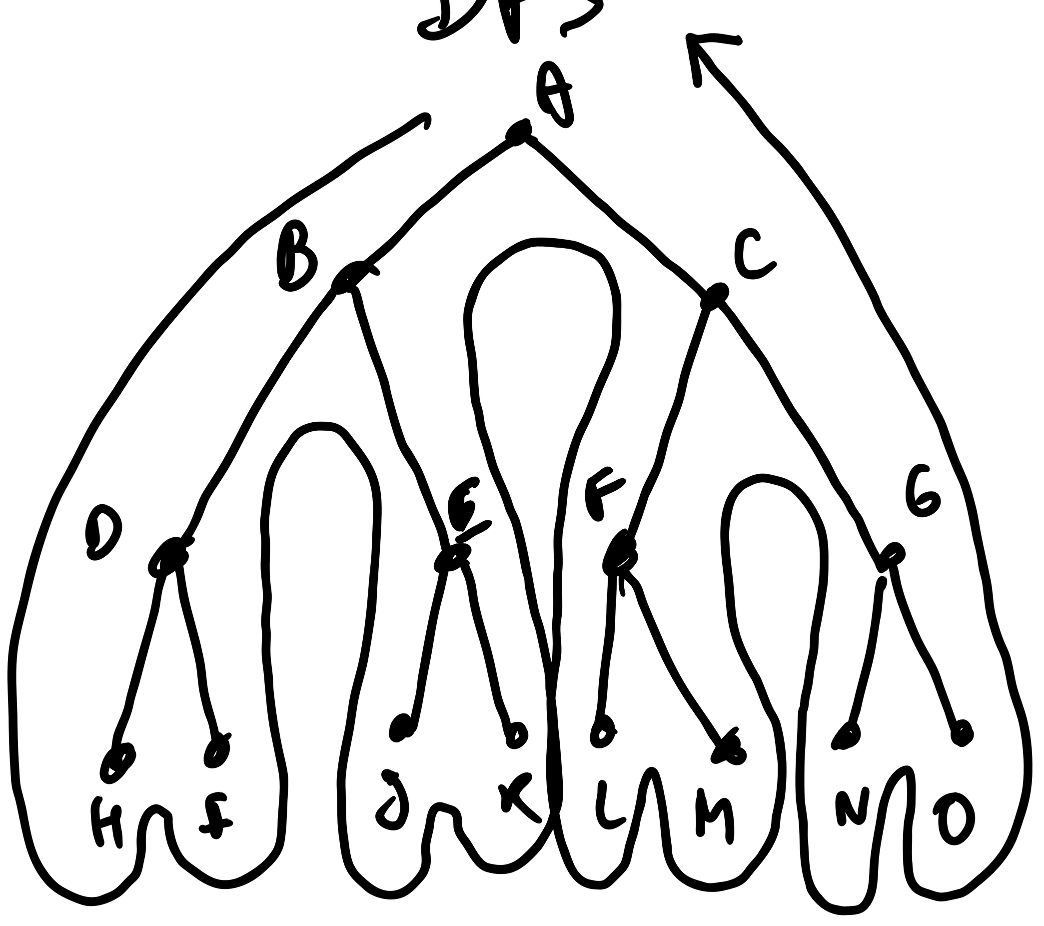
    créer pile (S)
    pour tous les u ∈ V :
        marqué[u] ← Faux
    empiler(S, r)
    tant que S ≠ ∅ :
        u ← dépiler (S)
        marqué [u] ← Vrai
        pour tous les uv ∈ E :
            Si marqué [v] = Faux :
                ~~marqué[v] ← Vrai~~
                empiler (S, v)



BFS



DFS



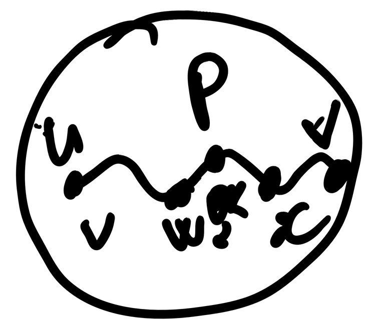
Version récursive de DFS

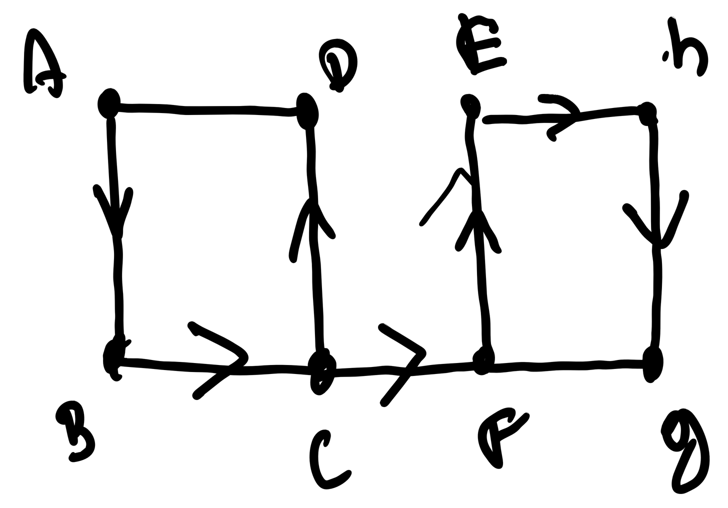
    procédure explorer (G, u) :
        marqué [u] ← Vrai
        pour tous uv ∈ E(G) :
            si marqué [v] = Faux :
                explorer (G, v)

    procédure DFS(G) :
        pour tous les u ∈ V(G)
            marqué [u] ← Faux
        pour tous les u ∈ V(G) :
            si marqué [u] = Faux :
                explorer (G, u)

Correction de DFS :

Supposons par l'absurde qu'à la fin de l'exécution de explorer(G,u), où G est un graphe connexe, il y a un sommet non marqué v. Soit P une chaîne de u à v. Soit w le dernier sommet marqué de P. Soit z le successeur de w dans P.
Contradiction, car explorer aurait "vu" z lorsque le sommet w est traité.

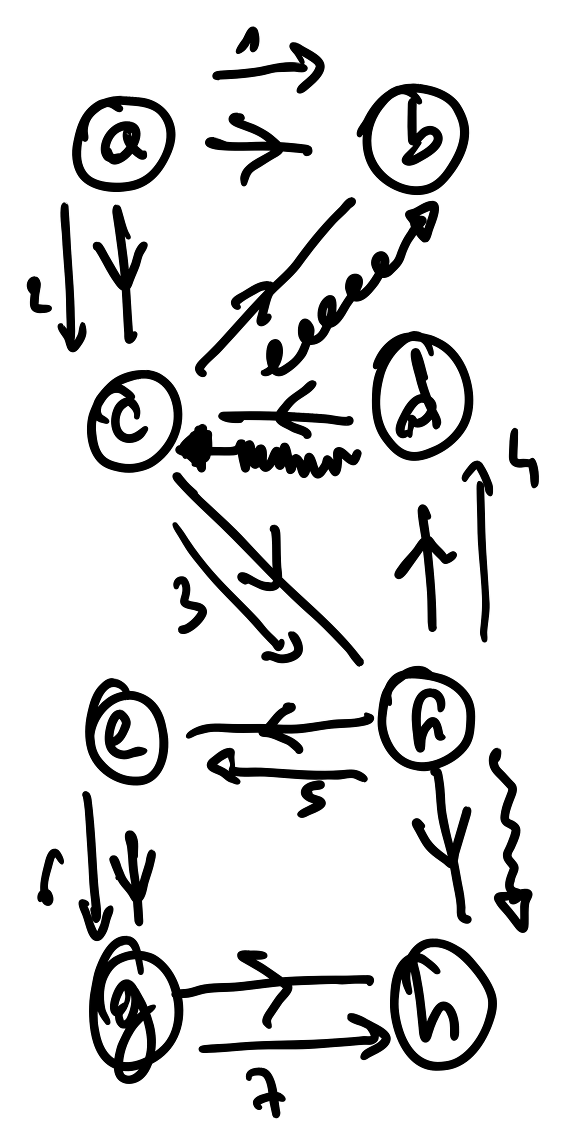




a, b, c, d, f, e, h, g

T est un arbre
- arêtes de l'arbre
- arêtes retour

DFS dans les graphes orientés



→ arcs de l'arbre DFS
→ arcs "avant" : arcs (u,v) tq u est un ancêtre de v (non parent)
→ arc "retour" : (u,v) où u est descendant de v.
→ arc "transverse" (le reste)

procédure DFS(G,u) :
marqué [u] ← Vrai
prévisite(u)
pour tous les (u,v) ∈ E
si marqué [v] = Faux
explorer (v)
postvisite (u)

procédure prévisite(u)
pre [u] ← t
t ← t+1

procédure postvisite(u)
post [u] ← t
t ← t+1

Soient u, v deux sommets quelconques. // théorème des parenthèses

Alors, soit (i) $[pre[u], post[u]] \subset [pre[v], post[v]]$

soit (ii) $[pre[u], post[u]] \supset [pre[v], post[v]]$

soit (iii) $[pre[u], post[u]] \cap [pre[v], post[v]] = \emptyset$

(i) $[_v \; [_u \;]_u \;]_v$

(ii) $[_u \; [_v \;]_v \;]_u$

(iii) $[_u \;]_u \; [_v \;]_v$

Soit (u,v) un arc de G

(i) Si (u,v) est un arc de l'arbre ou un arc "avant" alors on a $[_u \; [_v \;]_v \;]_u$

(ii) Si (u,v) est un arc retour, alors $[_v \; [_u \;]_u \;]_v$

(iii) Si (u,v) est un arc transverse, alors $[_v \;]_v \; [_u \;]_u$

Complexité de DFS

$O(n)$
$O(m)$
$O(n+m)$

$n = |V(G)|$ le nbr de sommets

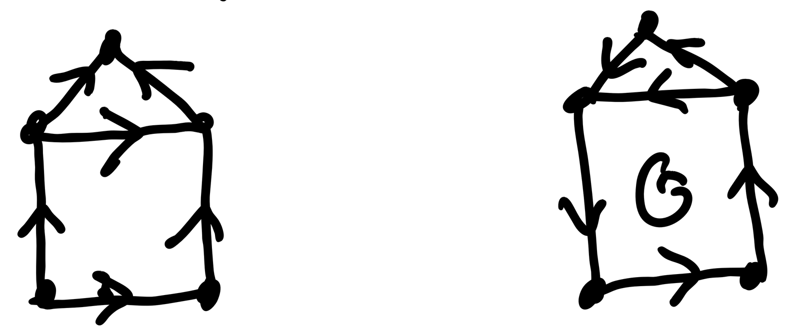
$m = |V(E)|$ le nbr d'arêtes

Graphe orientés acycliques

Exemple :

(DAG)

Directed acyclic graphs.



DAG

pas un DAG

Observation :

Un graphe orienté est acyclique (= ne contient pas de cycle orienté)
ssi le parcours en profondeur ne trouve aucun arc retour.

Démonstration

Supposons que G contient un arc retour (u,v)

v ancêtre de u

Donc, il existe un chemin P dans l'arbre DFS de v à u.

Supposons que G contient un cycle $C = (v_1, v_2, \dots, v_k, v_1)$

Soit v_i le premier sommet marqué. Tous les autres sommets de C sont atteignables à partir de v_i, et sont donc des descendants de v_i dans l'arbre DFS.

L'arc (v_{i-1}, v_i) est un arc retour (ou (v_k, v_1) si $i=1$).

Tri topologique d'un graphe orienté

ordre $v_1 < v_2 < v_3 < \dots < n$ st

si $(v_i, v_j) \in E(G)$, alors $v_i < v_j$