4 types of edges:

<u>tree edge</u>: edge $v \to u$,

explore(u) is called as a recursive call

the for-loop of

within explore(v).

solid edges in the figure

<u>forward edge</u>: edge $v \to u$, where in the (solid) tree:

u in subtree of v,

u is not a child of v.

in the figure: (A,F), (E,G)

<u>back edge</u>: edge $v \to u$, where in the (solid) tree:

v in subtree of u.

in the figure: (F,B), (D,A)

<u>cross edge</u>: all other edges.

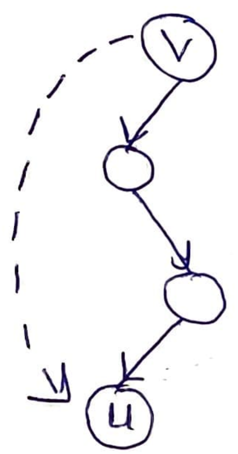
in the figure: (D,H), (H,G)

# How to decide the type of an edge?



**tree edges**: these are discovered during the algorithm.
Observe that for tree edge $(v,u)$: $pre(v) < pre(u) < post(u) < post(v)$.

**forward edges**: edge $(v,u)$ with



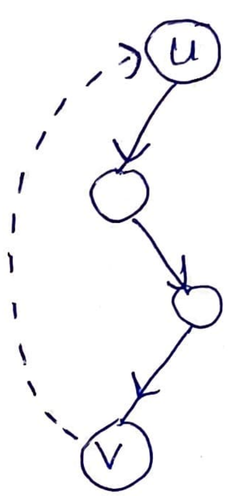

$(u,v) \neq$ tree edge

and

$pre(v) < pre(u) < post(u) < post(v)$

why: explore(u) starts and finishes within explore(v)

**back edges**: edge $(v,u)$ with




$pre(u) < pre(v) < post(v) < post(u)$

why: explore(v) starts and finishes within explore(u)

Cross edges: edge $(v,u)$ with

$$pre(u) < post(u) < pre(v) < post(v)$$

why: explore(u) finishes before explore(v) starts

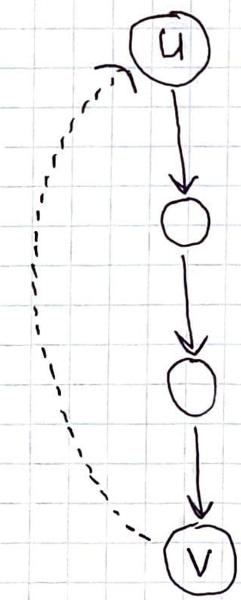
---

How to decide if a directed graph $G = (V, E)$ has a directed cycle:

Claim: G has a directed cycle $\Leftrightarrow$ DFS-forest has a back edge.

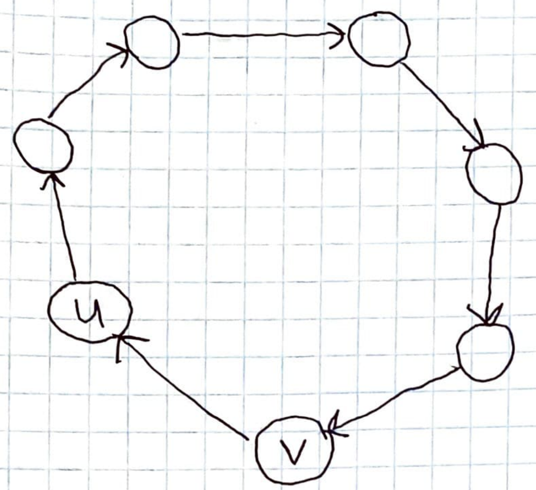
Proof: $\Leftarrow$ Assume $(v,u)$ is a back edge.

from page (83): in the solid tree, v is in the subtree of u.




The tree edges from u to v, plus edge $(v,u)$, form a directed cycle.

$\Rightarrow$ Assume $G$ has a directed cycle




This cycle contains an edge $(v,u)$ such that $post(v) < post(u)$.

Why: otherwise, $post(u) < post(v)$ for every edge $(v,u)$ on the cycle. Walk around the cycle, post-numbers get smaller and smaller and smaller ...
$\therefore$ Escher's Ascending and Descending $\lightning$

Take an edge $(v,u)$ on the cycle with $post(v) < post(u)$.

From page 84 : $(v,u)$ is not a tree edge and not a forward edge.

From top of page 85 : $(v,u)$ is not a cross edge.

$\therefore$ $(v,u)$ is back edge.

How to test if a directed graph is cyclic:

* Run DFS
* for each non-tree edge $(v,u)$, test if

$$pre(u) < pre(v) < post(v) < post(u)$$

if "yes" for at least one non-tree edge: cyclic.
if "no" for all non-tree edges: acyclic.

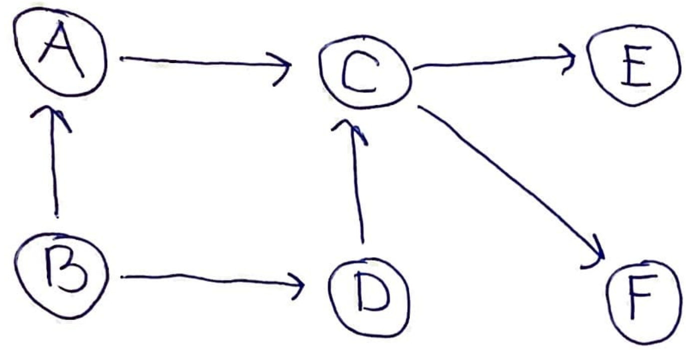
Running time: $O(|V|+|E|)$.

---

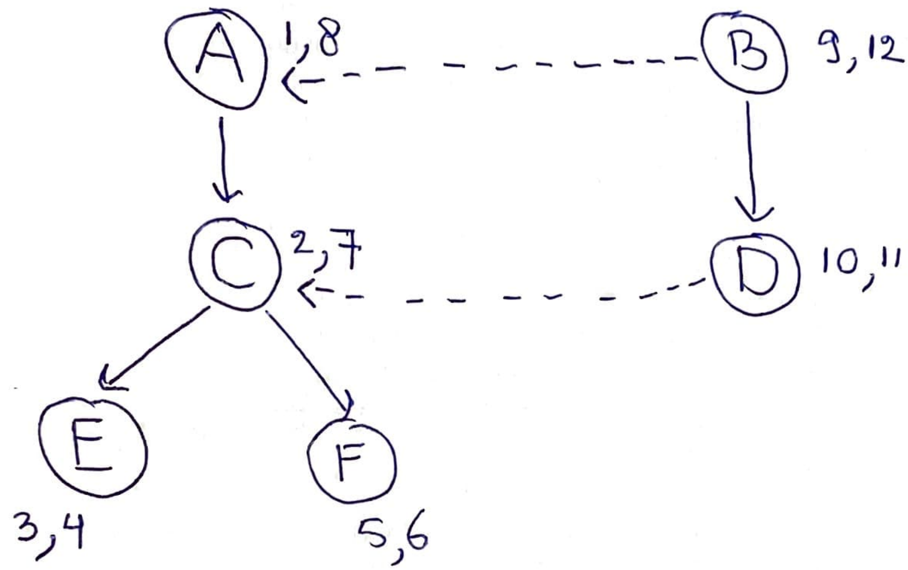
Assume G is acyclic. How to do topological sorting:

* Run DFS.
* Run Bucket-Sort to sort the vertices by post-number. [every post-number $\in \{2,3,4,\dots,2|V|\}$]
* obtain the topological sorting from the reverse sorted order.

Running time: $O(|V|+|E|)$.




Run DFS:

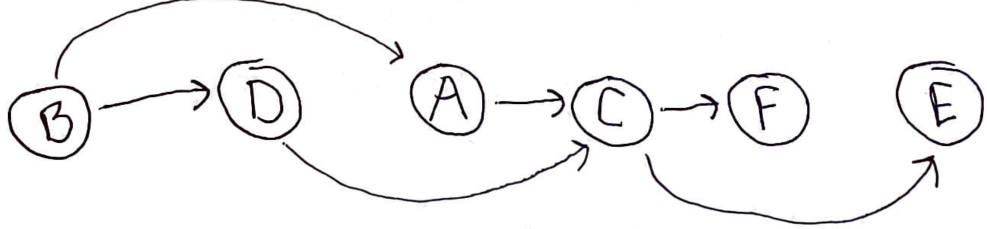



sort by post number: E, F, C, A, D, B

topological sort: reverse order:

B, D, A, C, F, E

1 2 3 4 5 6




all edges go from left to right

Correctness of algorithm at bottom of page 87:

Let $(v,u)$ be an edge of $G$.

To show: in topological sorting,

number of $v$ < number of $u$

Same as: $post(v) > post(u)$. ← To show.

Since $G$ is acyclic: $(v,u)$ is not a back edge (from claim on page 85)

$\therefore$ $(v,u)$ is a tree edge, or a forward edge, or a cross edge.

$\therefore$ From page 84 and top of page 85:

$$post(v) > post(u).$$

□