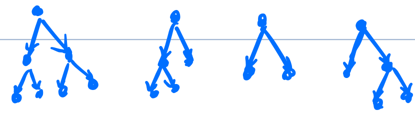
Recap: DFS: Pick an edge, explore everything going down that edge, then move on to the next edge

Gives a DFS tree / forest



Pseudo-Code

```
DFS-Visit(G, u)
    time += 1
    disc(u) = time
    color(u) = YELLOW
    For all v ∈ Adj[u]:
        If color(v) = WHITE:
            parent(v) = u
            DFS-Visit(G, v)
    color(u) = GREEN
    time += 1
    finish(u) = time
```

```
DFS(G)
    time = 0
    For all u ∈ V:
        color(u) = WHITE
        parent(u) = ⊥
    For all u ∈ V:
        If color(u) = WHITE:
            DFS-Visit(G, u)
```

Runtime: $\Theta(|V| + |E|)$

1. Edge Classification

- Tree Edge: Visit new vertex via edge (i.e. vertex was white)
- Forward Edge: node → descendant in tree
- Backward Edge: node → ancestor in tree
- Cross Edge: between two non-ancestor related vertices

How do we detect type of edge (u, v)?

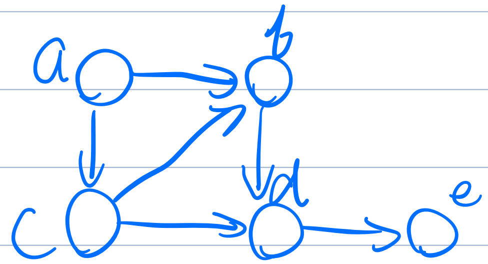
- Tree Edge: v was white when we explored this edge
- Forward Edge: - - - green - - - - -
- Backward Edge: - - - - yellow - - - -
- Cross Edge - - - green - - -

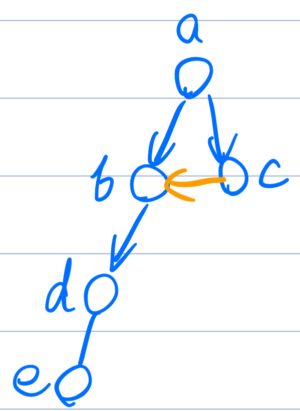
Cross Edge vs. Forward Edge?

Add discovery time (disc) and finish time (finish).

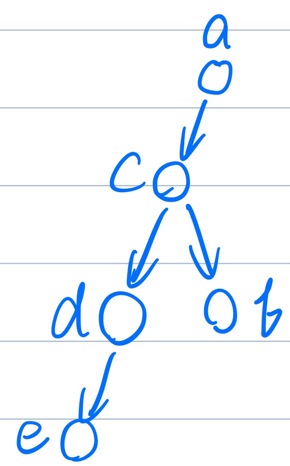
- Forward Edge: v was green & $disc(v) > disc(u)$
- Cross Edge: v was green & $disc(v) < disc(u)$ (and also $finish(v) < disc(u)$)

e.g.



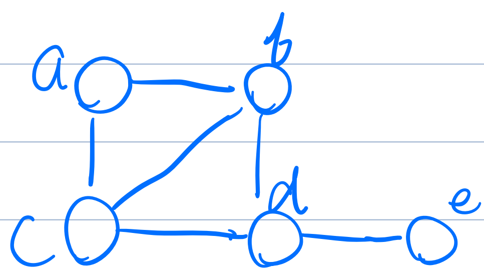


or



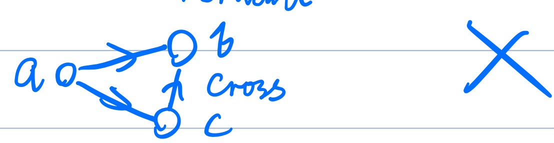
(c, b) is cross edge, but if we visit c, first, then (c, b) is tree edge

What if undirected?



We won't have

We also can't have



Undirected Graphs:

Thm: In DFS of an undirected graph G, every edge is either a tree edge or a backward edge, i.e. when we first explore $\{u,v\} \in E$ from u, v cannot be green.

Proof: If v is green, it means we finished exploring its neighbors, so we should have explored $\{u,v\}$ from v already. Contradicts with we first explore $\{u,v\}$ from u.

Corollary: An undirected graph is acyclic iff there are no backward edges.

Proof: If no backward edges, we only have tree edges, and trees are acyclic.
If we have a backward edge, then we have a cycle.

Directed Graphs:

Thm (White-Path Theorem): v is a descendent of u in the DFS forest iff at time $disc(u)$, there exists a path from u to v with only white vertices.

Proof Sketch:

"$\Rightarrow$": Let $u_0 = u, u_1, u_2, \dots, u_k = v$ be a path in the DFS forest (all edges (u_i, u_{i+1}) are tree edges). Then u_1 is discovered from u_0, u_2 from u_1, etc. Therefore,

$$\begin{tikzcd} u_0 = u \arrow[d, "T"] \\ u_1 \arrow[d, "T"] \\ \vdots \arrow[d, "T"] \\ u_k = v \end{tikzcd}$$

$$disc(u_k) > disc(u_{k-1}) > \cdots > disc(u_1) > disc(u_0)$$

So at time $disc(u_0) = disc(u)$, $u_1, u_2, \dots, u_k$ are all white.

"$\Leftarrow$": At time $disc(u_0) = disc(u)$, $u_1, \dots, u_k$ are all white.

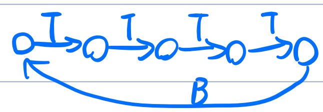
We have $disc(u_k) > disc(u_{k-1}) > \cdots > disc(u_1) > disc(u_0)$

$$finish(u_k) < finish(u_{k-1}) < \dots < finish(u_1) < finish(u_0)$$

Therefore, $u_k = v$ is discovered during the visit from $u_0 = u$, and there must a path from u to v in the DFS forest. $\#$

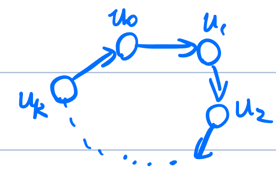
Corollary: A directed graph is acyclic iff DFS finds no backward edges.

Proof: We prove a directed graph is cyclic iff DFS finds a backward edge.

"$\Leftarrow$": 

Any backward edge immediately gives a cycle.

"$\Rightarrow$": Assume there is a cycle.



Let u_0 be the first vertex discovered in the cycle. So at $disc(u_0)$, $u_1, u_2, \ldots, u_k$ are all white. By white-path thm, u_k is descendent of u_0.

Then the edge (u_k, u_0) is a backward edge. #

Cycle detection? Run DFS and look for a backward edge!