Recap:

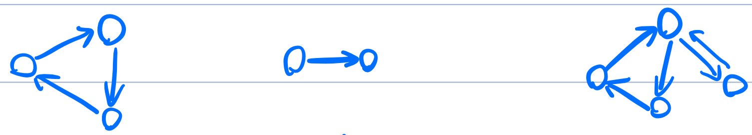
Cycle Detection: Run DFS and check for backward edges

Topological Sort: Run DFS, output vertices in decreasing finish time

See DAG, think topological sort!

1. Strongly Connected Components (SCC) (Kosaraju-Sharir '78/'81)

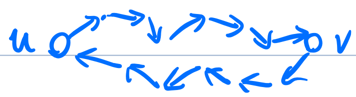
Strongly Connected: A directed graph $G=(V,E)$ is strongly connected if $\forall u,v \in V$, $u \neq v$, $\exists$ path from u to v (and from v to u).



✓ ✗ ✓

Claim: A strongly connected graph must contain a cycle?

✓. Can be a single vertex. Otherwise, must contain a cycle.

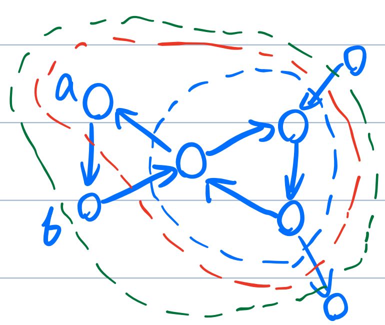


path u to v + path v to u = cycle!

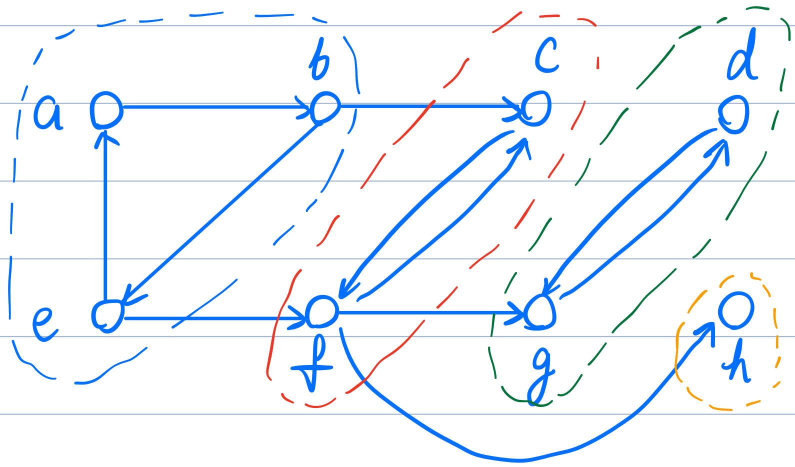
$G' = (\text{subset } V' \subseteq V, \text{ edges with both ends in } V')$

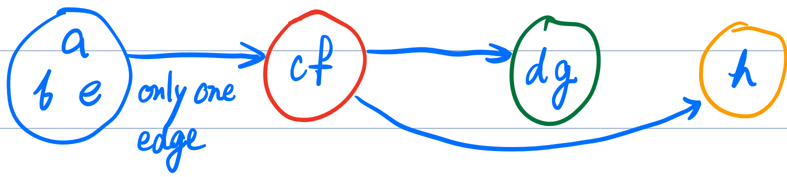
Strongly Connected Component (SCC): A subgraph of G that is strongly connected and maximal.

We cannot add any other vertices to V' and have it still strongly connected.



* Vertices, not vertex: cannot add a or b alone, but can add a and b together

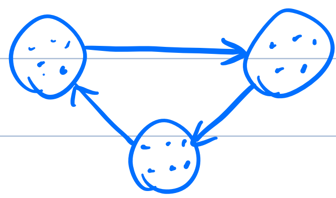




G^{scc}: Component graph

Claim: The component graph is a DAG.

Proof: A cycle forms a larger SCC. Contradiction!



Consider running DFS on G.

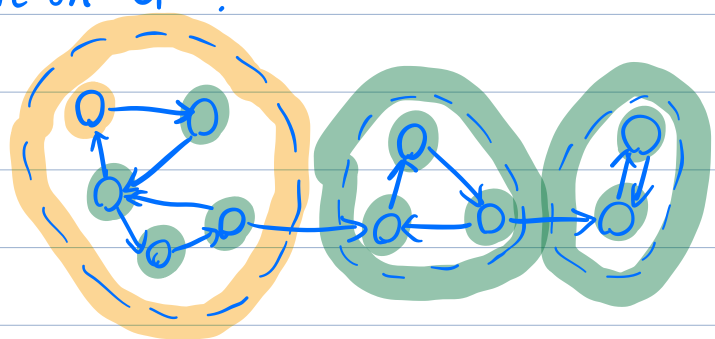
We can pretend that this DFS takes place in G^{scc} in the following way:

- Mark a component discovered (YELLOW) when any of its vertices is discovered
- Mark a component finished (GREEN) when all of its vertices are finished
- A component starts WHITE, then YELLOW, then GREEN

Key Lemma: the "virtual" walk above on G^{SCC} induced by DFS on G is a valid DFS exploration of G^{SCC}.

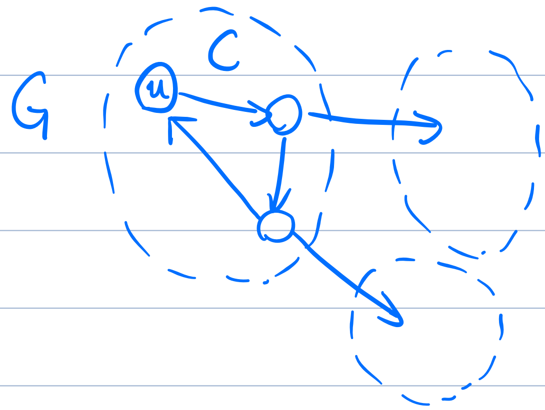
Proof Idea: When a component C is first discovered, say by a call to DFS-Visit on some $u \in C$, when that call returns, all vertices in C as well as in all components reachable from C by white path, are finished, i.e. GREEN, just like a real DFS visit on G^{SCC}.

#

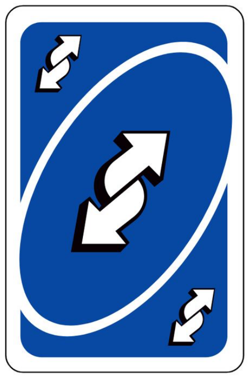


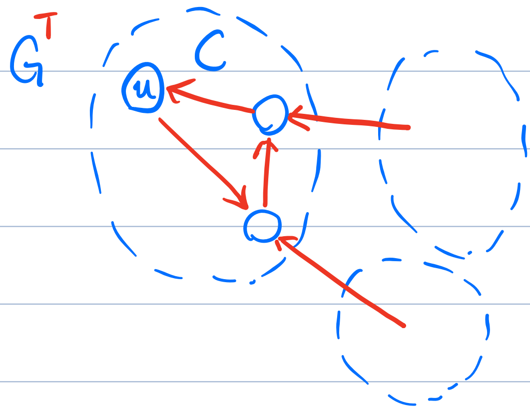
If we have $C \longrightarrow C'$ in G^{SCC}, then finish$(C) >$ finish(C'). Therefore, the component with the largest finish time must be the first in the topological sort of G^{SCC}.

Observation: the last vertex to finish is the left-most component (in the topological sort of G^{SCC})



Let's say u is the last vertex to finish.
C is the left-most component.
How do we find out what vertices are in C?





We run DFS-Visit(G^T, u).
G^T is a copy of G, with all edges reversed.

SCC(G):

1. Run DFS(G) to compute finish(u) for all $u \in V$.
2. Run DFS(G^T), but in the main outer loop, consider vertice in order of decreasing finish 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)
```

here! Loop by decreasing finish time.

3. DFS in step 2 gives a DFS forest.
Output the vertices of each tree in the forest as a separate SCC.

<u>Runtime</u>: $\Theta(|V|+|E|)$

$\boxed{\text{If you see directed graph, think SCC!}}$