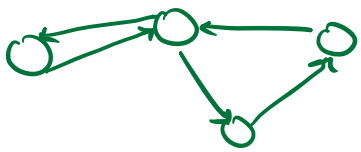
Strongly Connected Components

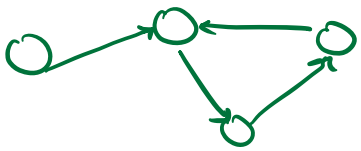
Def: A directed graph $G = (V, E)$ is strongly connected if

for all $v, w \in V$:

there exists a path from v to w and w to v



strongly connected



not strongly connected

Finding SCC's

1. Straightforward DFS soln

```
SCC = []
Foreach u ∈ V:
    Run DFS from u
    For each w reachable from u:
        if w is in an SCC already found:
            if DFS from w shows u is reachable:
                add u to SCC[v]
                break
    If no break:
        create new SCC containing only u
```

At least $|V|$ DFS calls that take time $O(|V|)$

$\Rightarrow$ runtime $\Omega(|V|^2)$

2. Efficient $O(|V|)$ soln

DFS, create DFS forest (choose starting vertices in order, keep track of finishing times)

Reverse all edges in the graph

Do DFS again to create another DFS forest

(order nodes in reverse order of previous finishing times)

SCCs are different trees in second DFS forest

Why does this work?

Lemma 1: The SCC graph (where each SCC is replaced w/ a node) is a directed acyclic graph (DAG)

Why? If not, could merge two or more nodes into a single SCC

Start and finishing times

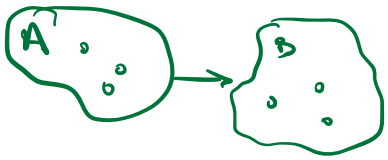
Def':

finish time': largest finish time of any node in SCC

start time': smallest finish time of any node in SCC

Lemma 2: If there is an edge $A \to B$ in the SCC DAG, then A.finish > B.finish

Proof of Lemma 2:



Case 1. Reached A before B in first DFS

$\Rightarrow$ Suppose y has largest finish time in B ($B.finish = y.finish$)

$\Rightarrow$ Suppose z was discovered first in A ($A.finish \geq z.finish$)

$\Rightarrow$ Will discover y via z $\Rightarrow$ y is descendant of z in forest

$\Rightarrow$ $A.finish > B.finish$

Case 2: Reached B before A in first DFS

$\Rightarrow$ No paths from B to A (no cycles)

$\Rightarrow$ A explored later after restarting DFS

$\Rightarrow$ $A.finish > B.finish$

Corollary of Lemma 2: If there is an edge $B \to A$ in reversed SCC DAG, then $A.finish > B.finish$

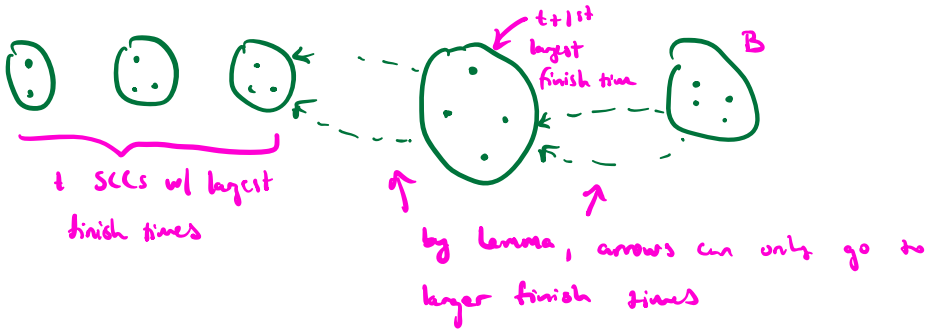
Proof of algo correctness by induction

IH: first t trees found in second DFS are t SCC's w/ largest finish time

BC: 0 (vacuous)

IS: assume first t trees are last-finishing SCCs

Consider $(t+1)^{th}$ tree, suppose root x, suppose $x \in$ SCC A



Then A.finish > B.finish for all remaining SCCs B

Then no edges leaving A in remaining SCC DAG

Then DFS started at x reaches exactly A

(nothing more due to reachability, nothing less due to A being an SCC)

$\Rightarrow$ So $t+1$'th tree has $t+1$'th largest finish time

BC proved, IS proved $\Rightarrow$ done!