Linear time $O(|V| + |E|)$

\[
\begin{bmatrix}
\ldots
\end{bmatrix}
\]

\begin{align*}
5 & 6 & 7 & 8 \\
p_{\text{pre}(u)} & & & \\
p_{\text{post}(u)} & & &
\end{align*}

1. Any descendant of $u$ has its interval as a subset of $[\text{pre}(u), \text{post}(u)]$

2. $\left[\ldots \right] \cup \left[\ldots \right] = \emptyset$

$x$ and $y$ are sibling nodes.
x and y are sibling nodes, 
\[ \exists y \] 
share a common ancestor

---

**How to detect cycles?**

1. Use DFS to create the $[pre(x), post(x)]$ intervals
   \[ O(|V| + |E|) \]
   \[ \text{Interval}(x) \]

2. \( \forall \text{edge} \ e \in E \)
   \[ e = (u, v) \]
   \[ \text{if} \ Interval \ [u] \subseteq \text{Interval} \ (v) \]
   \[ O(|E|) \]
   \[ \text{then output} \ "\text{cycle found}" \]

---

Directed Graph $\xrightarrow{\text{how to detect cycles}}$ Directed Acyclic Graph (DAG) $\xrightarrow{\text{deadlocks}}$

Directed Graph

Directed Acyclic Graph (DAG)

Scheduling

Linearize the graph: for every node $u$,

- **topological**
- its ancestors must
  \[ \ldots \]
Topological Sorting

### In-degree

![Diagram of a directed acyclic graph (DAG) with nodes A, B, C, D, E, F and their in-degrees]

1. Any DAG has at least one source node with no incoming edges.
2. Any DAG has at least one sink node with no outgoing edges.

#### Algorithm 1 (Topo sort)

\[ TS(G) \]

1. Find a source, output it.
2. Delete the source and its outgoing edges, resulting in \( G' \).
3. Call \( TS(G') \).

```plaintext
source queue: B A D F
output: B A D C E F
```

\[ \text{Time: } O(V + E) \]

---

1. Compute all in-degrees \( O(V + E) \).
2. Any node with in-degree 0 is a source, add it to the queue.
3. For all neighbors of a source node.
for all neighbors of a source node
    decrement the indegree
if indegree = 0 add to queue

$O(M + E)$

Topological sort using DFS numbering

If you list the nodes
   in decreasing post(x)
order, then we get a
   topological sort!

B D A C E

any node x that can reach y

will have a higher post number

$\text{post}(x) > \text{post}(y)$
$O(|V| + |E|)$ time.

**Strongly Connected Components of a Directed Graph**

$scc(G)$

$scc_1, scc_2, scc_3$

$scc$: there is a directed path between each pair of vertices and the set is maximal.

**Directed Graph**

**Undirected graph**

Connected component is a maximal set of reachable nodes.

$\implies$

Yielded representation

$\overrightarrow{DA}\overrightarrow{G}$

over $Scc_5$

Black edges: $G^R$

lecture12 Page 5
Alg: SCC(G)

Input: G is a directed graph

Output: the list of SCCs.

1. Create the reverse graph $G^R$

2. DFS numbering on $G^R$

3. Identify source node by the largest post number (in $G$)

4. Any source in $G^R$ is a sink in $G$