DFS numbering

pre-order number
post-order number

DFS numbering

1) where we start
2) how we visit the outgoing edges

Containment of intervals indicate ancestor-descendant relationships

Detecting cycle

DA6 -> Directed Acyclic Graph

Given a directed graph, is it a DAG or not?

-> Does it have a cycle or not?
Given a directed graph, does it have a cycle or not?

Observation: If there is a cycle, then there exist nodes u and v such that

\[ [s_u, e_u] \leq [s_v, e_v] \]

such that

\[ \text{Interval of } u \subseteq \text{Interval of } v \]

If at v, we find a link to some vertex u such that Interval of v \subseteq Interval of u that has already been visited,

then \( v \) is a cycle.
Alg to Detect Cycles:
1) Do DFS intervals:\[O(M+|E|)\] linear time
2) \[\forall v \in V\]
   for all \[x \in \text{neighbor}\(s\) of \text{v}\]:
   \text{If Interval} [x] \supseteq \text{Interval} [v] \text{ cycle found}:
   \[
   \begin{array}{c|cccc}
   0 & 1 & 2 & 4 & 8 \\
   \hline
   [2, 5] & [7, 10] & \hline
   \end{array}
   \]
   \[O(M+|E|)\] worse case

Topological sort
Dependency graph
Dependency graph

All of the "pre-requisites" of a node must come before that node in the ordering.

Topo Sort:

1) Do a DFS numbering

2) Output vertices in reverse order of the post-order link

This is a topological sort

\[ O(V + E + V \log V) \]

\[ O(V + E) \]

\[ O(E + V \log V) \]
\begin{equation*}
\mathcal{O} \left( \frac{1}{e} + \frac{1}{|V| \log |V|} \right)
\end{equation*}

\begin{equation*}
\mathcal{O} \left( |V| + |E| \right)
\end{equation*}

Post order list:

- \(e=3\)
- \(f=5\)
- \(c=6\)
- \(a=8\)
- \(d=11\)
- \(b=12\)

"Sorting is for free."

\underline{Proof of correctness:}

\begin{itemize}
\item \textit{Interval} \([a, b]\)
\item Contains all reachable node's intervals
\item \(Post(u)\) must be larger than all other nodes that are reachable from \(u\)
\end{itemize}
1) all descendants of \( u \) have smaller post value

2) all ancestors of \( u \) have layer post value

**Strongly Connected Components (SCC)**

Directed graph (with cycles)

SCC: maximal set of vertices that are mutually reachable from each other

\( \{x, y\} \in \text{SCC} \)

\( x \) can reach \( y \) \( \iff \) \( y \) can reach \( x \) \( \iff \) mutually reachable

we cannot add another vertex to the SCC
Directed graph has an inherent **DAG structure**

over **SCC**

**Directed graph**

Any directed graph is a **DAG** over its strongly connected components

**SCC Algorithm:**

Given a directed graph \( G = (V, E) \)

output all **SCCs**

\[ O(\|V| + \|E|) \]

1) Identify successively all "sink" nodes

2) From each sink, do a DFS to extract the SCC

\[ \text{Remove it} \rightarrow \text{mark all vertices as visited} \]

A source vertex in a directed graph has the highest
1) Do DFS numbering on reverse graph $G^R$

2) Just traverse in decreasing order of $\text{post}(v)$ in $G$ and find all reachable vertices from $v$

Every such set is a SCC.

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