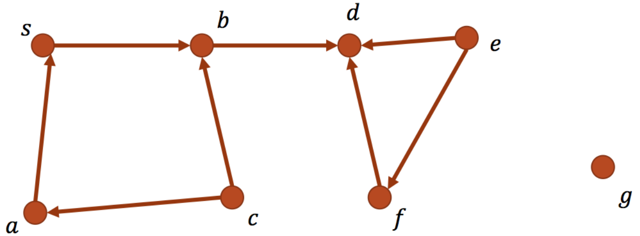
COL106: Data Structures and Algorithms

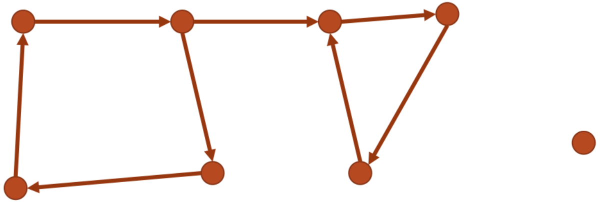
Ragesh Jaiswal, IIT Delhi

- Question: Given a directed graph that contains a cycle. Is topological ordering possible?
- Question: Given a DAG. Is topological ordering possible? If so give an algorithm that outputs one such order. What is the running time?



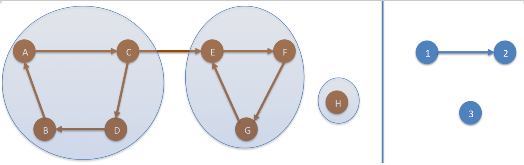
Problem

Given a directed graph $G = (V, E)$, output all the strongly connected components of G.



Problem

Given a directed graph $G = (V, E)$, output all the strongly connected components of G.



- Question: Given a directed graph G, consider a graph G^{scc} defined as follows:
  - There is a vertex in G^{scc} for each strongly connected component of G. That is, if $A_1, A_2, ..., A_k$ are k vertex sets of different strongly connected components of G, then G^{scc} has k vertices $1, ..., k$
  - There is a directed edge from i to j in G^{scc} iff there are $u \in A_i$ and $v \in A_j$ such that there is a directed edge from u to v in G.

  What kind of graph is G^{scc}?

Problem

Given a directed graph $G = (V, E)$, output all the strongly connected components of G.

- Given a directed graph G, consider a graph G^{scc} defined as follows:
  - There is a vertex in G^{scc} for each strongly connected component of G. That is, if $A_1, A_2, ..., A_k$ are k vertex sets of different strongly connected components of G, then G^{scc} has k vertices $1, ..., k$
  - There is a directed edge from i to j in G^{scc} iff there are $u \in A_i$ and $v \in A_j$ such that there is a directed edge from u to v in G.
- Claim: For any directed graph G, the graph G^{scc} constructed as above is always a DAG.

- Suppose during GraphDFS(G), we record:
  - the time at which a node v is discovered as the *start time* of v denoted by $start(v)$, and
  - the time at which we are done exploring the neighborhood of v (or when the recursive call returns) as the *finish time* of v denoted by $finish(v)$.
- The following procedure records these times.

Algorithm

- $time \leftarrow 0$

GraphDFS-with-start-finish(G)

- While there is an "unexplored" vertex u
- DFS-time(u)

DFS-time(u)

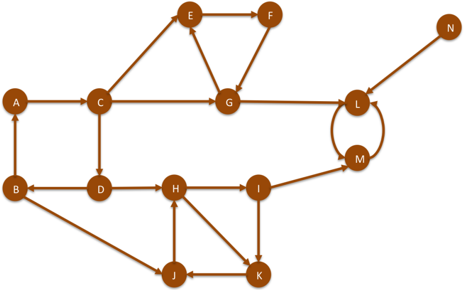
- Mark u as "explored" and set $start(u) \leftarrow ++time$
- While there is an "unexplored" neighbor v of u
  - DFS-time(v)
- $finish(u) \leftarrow ++time$

Algorithm

- $time \leftarrow 0$

`GraphDFS-with-start-finish`(G)
- While there is an "unexplored" vertex u
- `DFS-time`(u)

`DFS-time`(u)
- Mark u as "explored" and set $start(u) \leftarrow ++\, time$
- While there is an "unexplored" neighbor v of u
  - `DFS-time`(v)
- $finish(u) \leftarrow ++\, time$



Algorithm

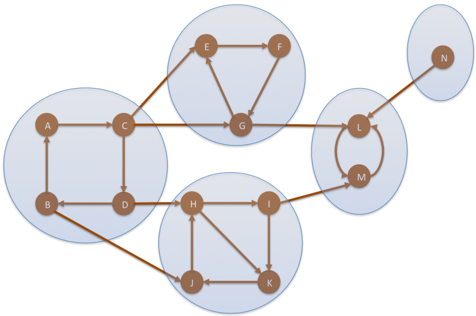
- $time \leftarrow 0$

`GraphDFS-with-start-finish(`G`)`

- While there is an "unexplored" vertex u
- `DFS-time(`u`)`

`DFS-time(`u`)`

- Mark u as "explored" and set $start(u) \leftarrow ++time$
- While there is an "unexplored" neighbor v of u
  - `DFS-time(`v`)`
- $finish(u) \leftarrow ++time$



Algorithm

- $time \leftarrow 0$

`GraphDFS-with-start-finish(`G`)`
- While there is an "unexplored" vertex u
- `DFS-time(`u`)`

`DFS-time(`u`)`
- Mark u as "explored" and set $start(u) \leftarrow ++time$
- While there is an "unexplored" neighbor v of u
  - `DFS-time(`v`)`
- $finish(u) \leftarrow ++time$

- Let $V_1, V_2, ..., V_k$ be the k vertex sets of strongly connected components of G.
- Consider G^{scc} where the vertices are labeled $1, 2, ..., k$.
- Claim 1: If there is a directed edge from node i to node j in G^{scc}, then the highest finish time among vertices in V_i is bigger than the highest finish time among vertices in V_j, when `Graph-DFS-with-start-finish(`G`)` time executed.

End