CSE 5311 Notes 13: Depth-First Search

SEARCH STATUS

Vertex colors:

White - undiscovered

Gray - discovered, on stack

Black - all outgoing edges processed (and returned from recursion)

Discovery time (count) - time of changing white ⇒ gray

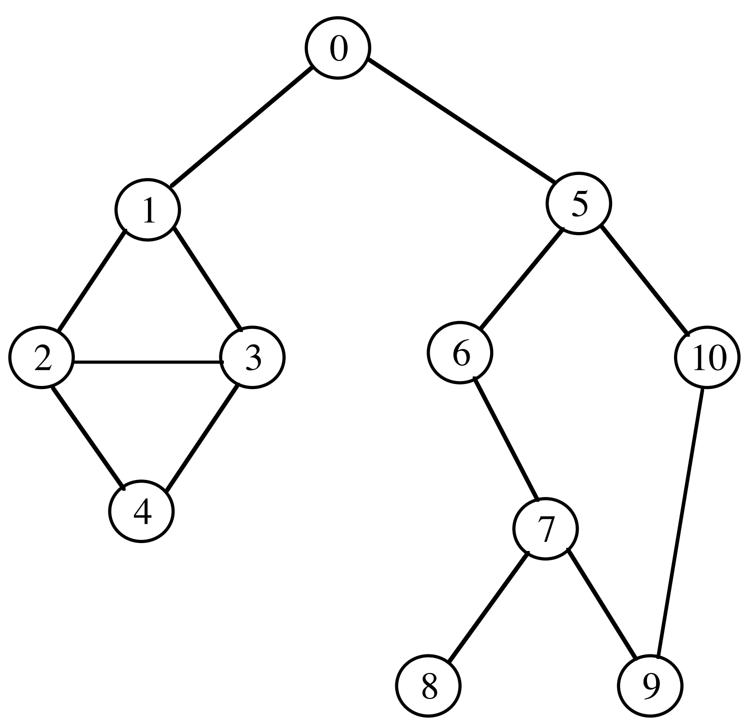
Finish time - time of changing gray ⇒ black

UNDIRECTED

Two edge types:

Tree ⇒ To undiscovered vertex

Back ⇒ To ancestor in DFS tree



DIRECTED

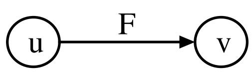
Four edge types:

Tree ⇒ To undiscovered vertex (white)

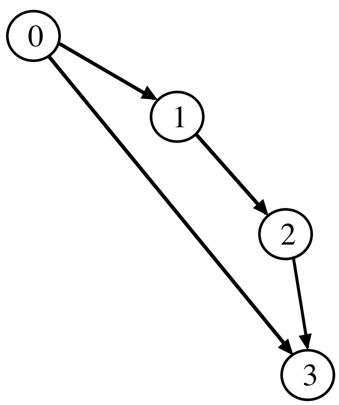
Back ⇒ To ancestor in DFS tree (gray)

Edges to black vertex

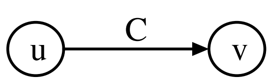
1. Forward



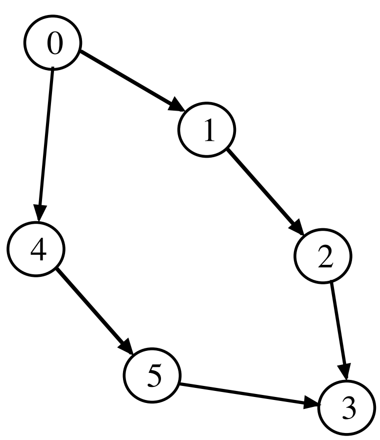
discovery(u) < discovery(v)

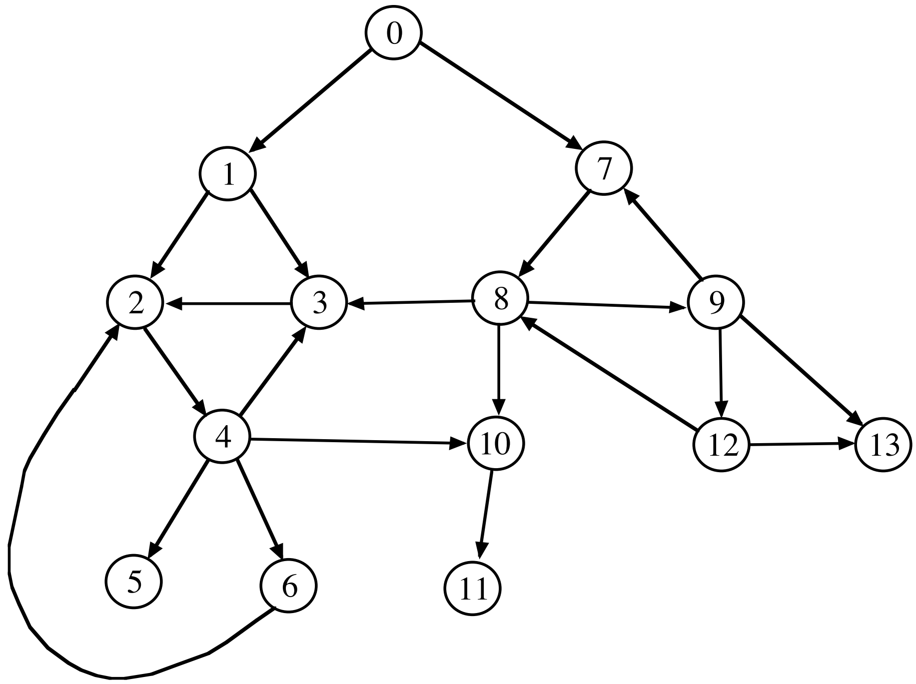


2. Cross



discovery(u) > discovery(v)





TOPOLOGICAL SORT OF A DIRECTED GRAPH (review)

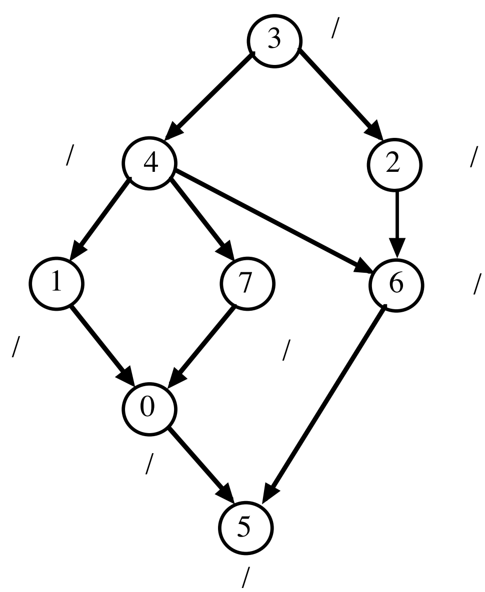
Linear ordering of all vertices in a graph.

Vertex x precedes y in ordering if there is a path from x to y in graph.

Apply DFS:

1. Back edge ⇔ graph has a cycle (no topological ordering).

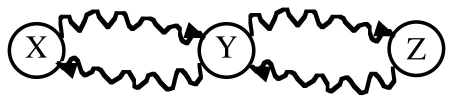
2. When vertex turns black, insert at beginning of ordering (ordering is reverse of finish times).

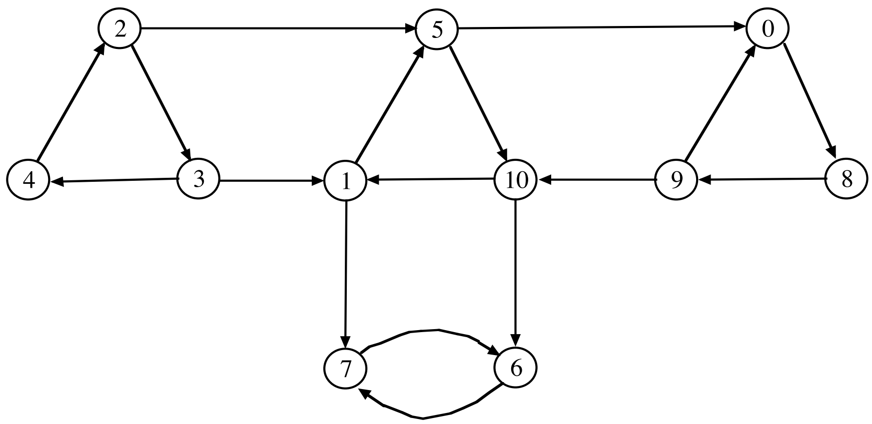


3 4 7 2 6 1 0 5

STRONGLY CONNECTED COMPONENTS (review)

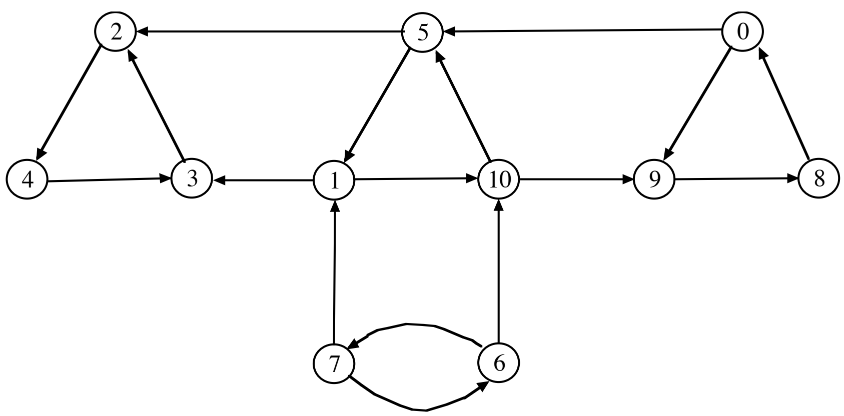
Equivalence Relation – definition (reflexive, symmetric, transitive)





1. Perform DFS. When vertex turns black ⇒ insert at beginning of list.

2. Reverse edges.



3. Perform DFS, but each restart chooses the first white vertex in list from 1. Vertices discovered within the same restart are in the same strong component.

Observation: If there is a path from x to y and no path from y to x, then finish(x) > finish(y).

This implies that the reverse edge (y, x) corresponding to an original edge (x, y) without a "return path" will be a cross edge during 2nd DFS. y will be in a SCC that has already been found.

Takes Θ(V + E) time.

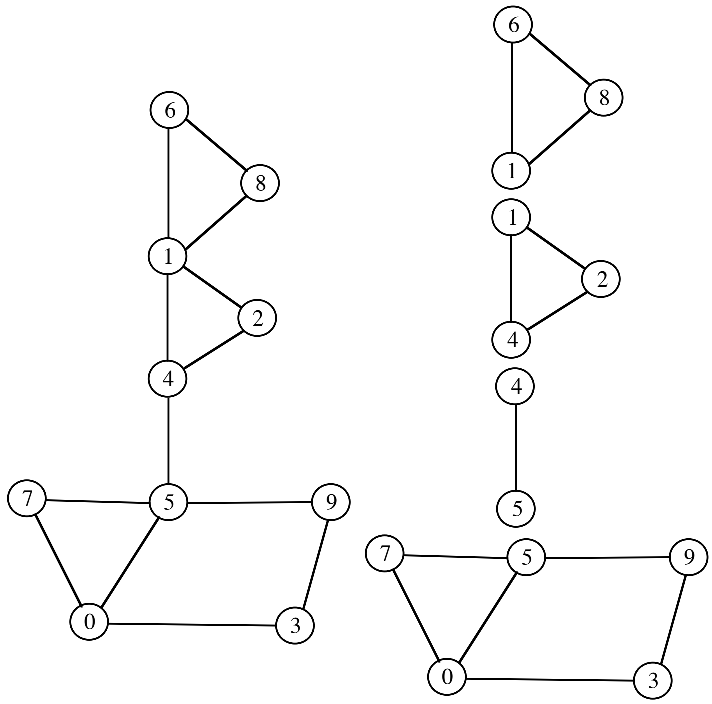
BICONNECTED COMPONENTS

An undirected graph is biconnected if removing any single vertex does not disconnect the graph.

Bicomponent = a maximal subgraph that is biconnected

The bicomponents of a graph are characterized by the articulation points.

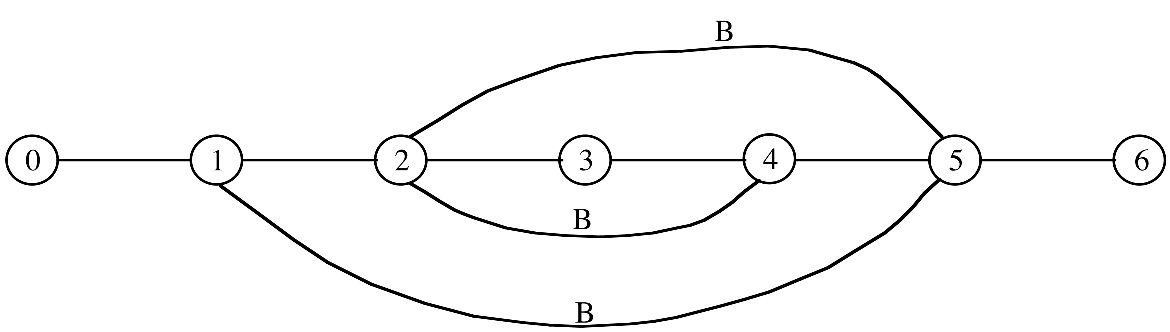
Any two vertices in a bicomponent (with > two vertices) are in a simple cycle.



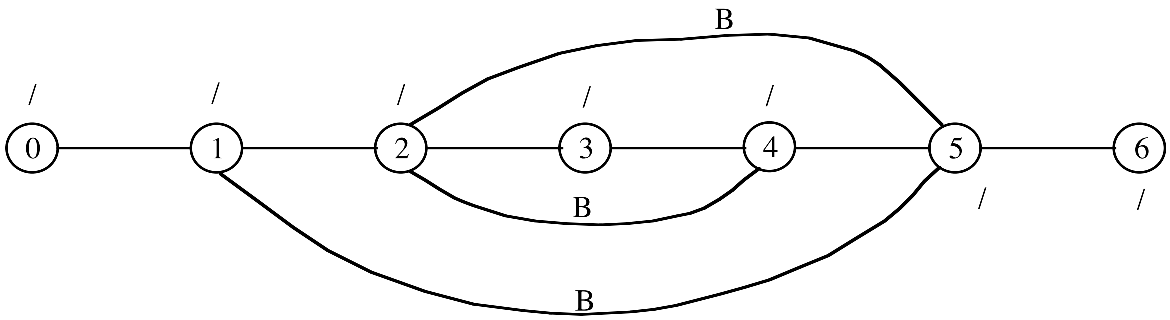
Use of DFS

If a DFS subtree does not have a back edge to a proper ancestor of the root, then

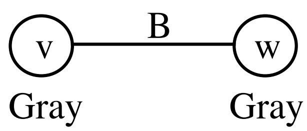
1. the root is an articulation point and
2. all edges since the last articulation point are in the same bicomponent. (Use a stack.)



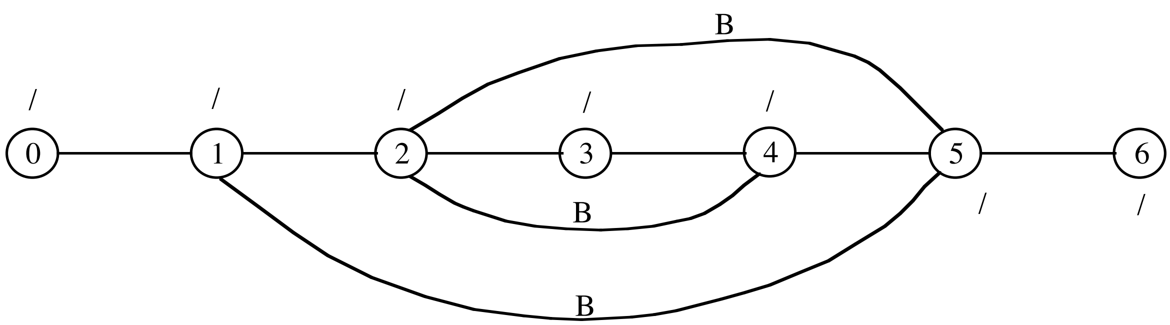
1. Use usual DFS to number vertices with discovery time. (Push each edge on stack.)



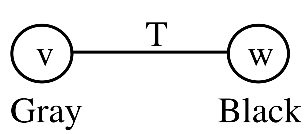
2. For each vertex, maintain the *earliest* vertex "cycled back to". (Initialize to discovery time.)



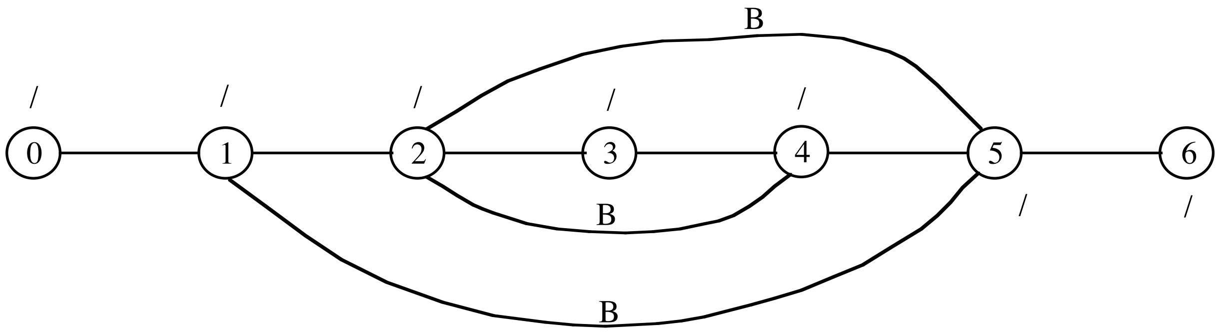

```
back[v] = min(back[v],discovery[w]);
```



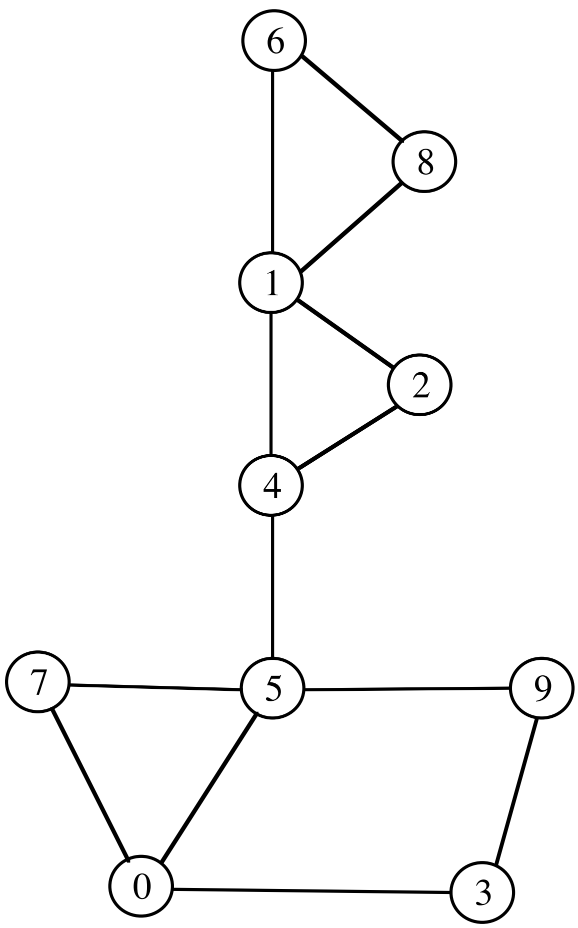
3. When returning on a tree edge (v, w), check for alternate path to vertex before v.

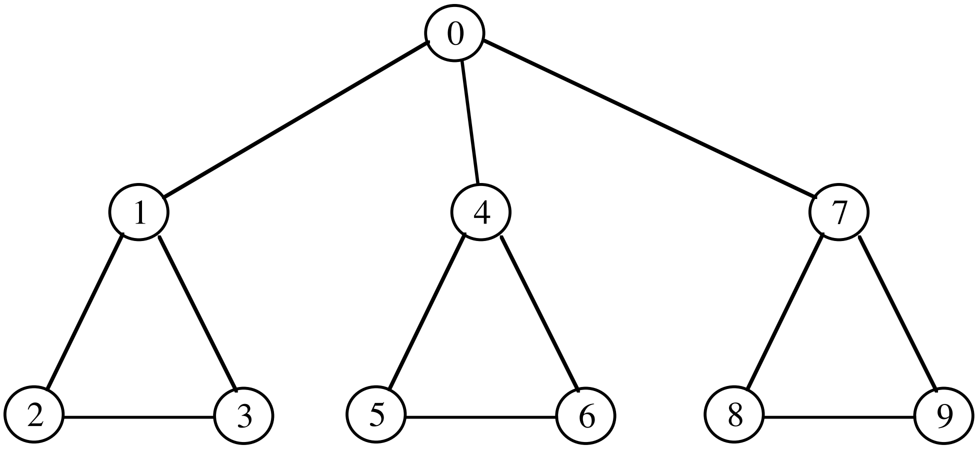



```
if back[w] >= discovery[v]
     v is an articulation point
     Pop edges until (v, w) has been popped
else
     back[v] = min(back[v],back[w]);
```



<u>Stack</u>





Since 0 started the DFS and it has ______________________________________, it is an articulation point.

What if {3, 5} and {6, 8} are present?