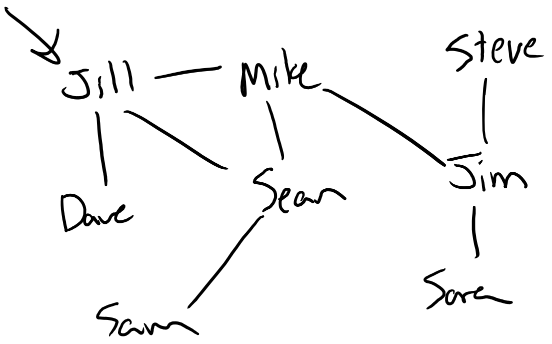
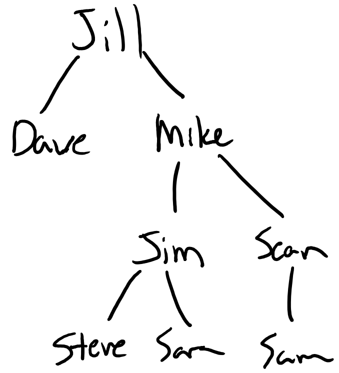
2018-04-10 DFS Articulation Trees

Tuesday, April 10, 2018 8:57 AM

- Trees can also be built from a DFS

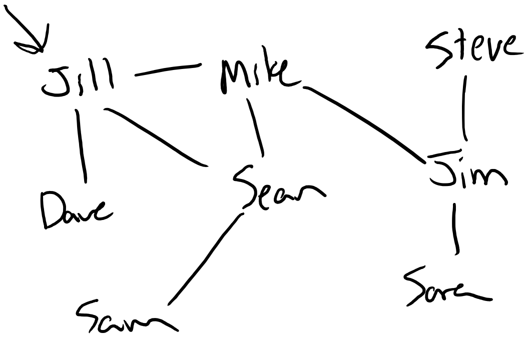


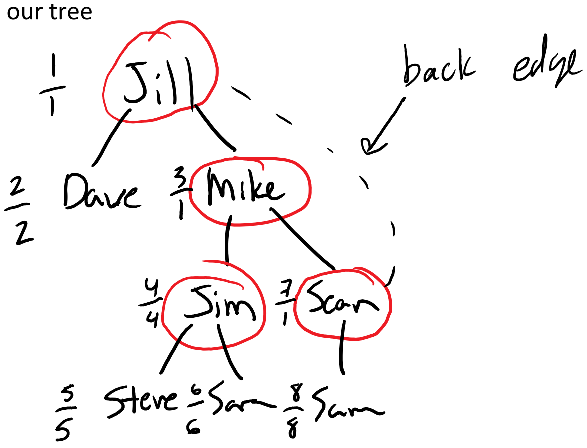


DFS trees allow us to find weak points in a graph (articulation points).

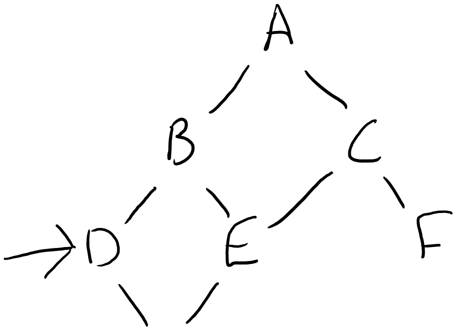
- In social graph, if one person were to go away (die), who would no longer be friends?
- Network analysis: If this switch or router goes down, some people will not have Internet access
- Traffic patterns: In the case of a natural disaster, will some people get stranded?
- Military conflicts: what bridge must we hold? If we destroy this bridge, the enemy can't get to us as easily

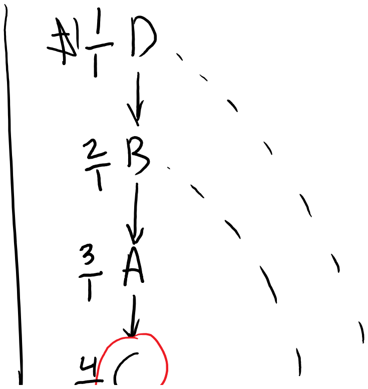
- To find an articulation point using a DFS tree, we must incorporate "back edges" into our tree.
  - Defined: A back edge is an edge in the graph that we ***could*** have taken but didn't because we've already visited that node in our DFS traversal.
- We represent back edges using a dotted line in our tree

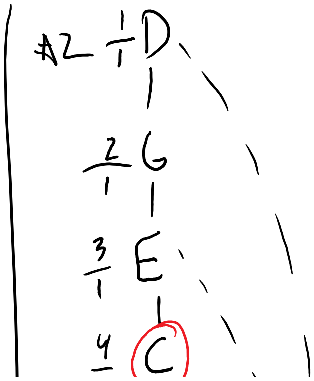


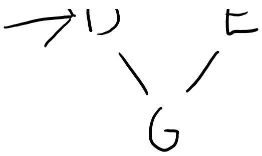


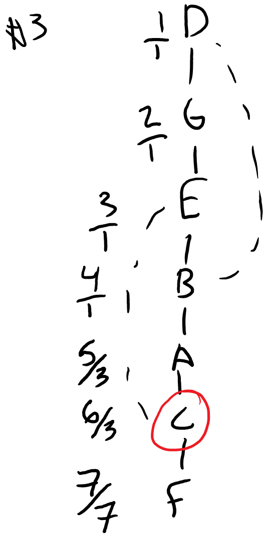
Class Exercise: Draw a DFS articulation tree for the following graph (starting at D)

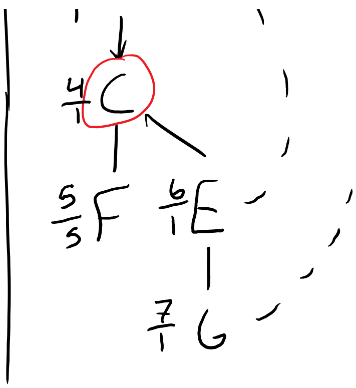


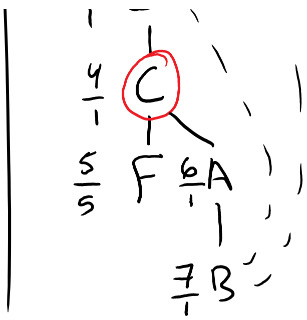








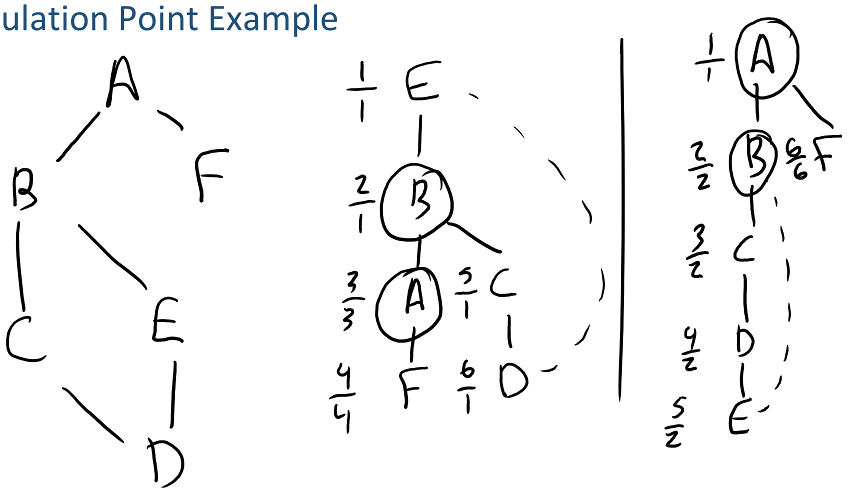




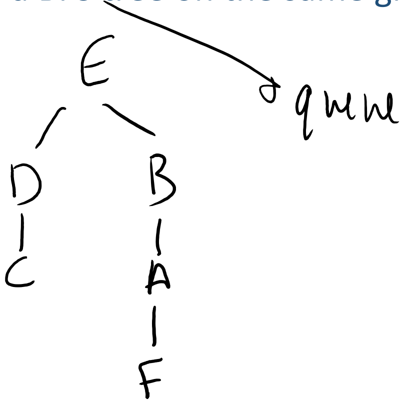
Articulation Point Algorithm

1. Give each node an ID value based on the order that it was visited (start at ID #1 -> root)
2. Next, find the lowest ID of the node that can be reached by taking zero or more forward edges (solid lines) and ***up to one*** back edge (dotted line).
3. We represent this information as a fraction for convenience: ID / LOW VALUE
4. A node is an articulation point:
   a. At the root ***if and only if*** the root has two or more children
   b. At all other nodes when the node's child's LOW value >= its ID value

Full Articulation Point Example



Practice: Draw a BFS tree on the same graph starting at E



Efficiency of Articulation Point Algorithm

N 1. Give each node an ID value based on the order that it was visited (start at ID #1 -> root)

N^2 2. Next, find the lowest ID of the node that can be reached by taking zero or more forward edges (solid lines) and ***up to one*** back edge (dotted line).

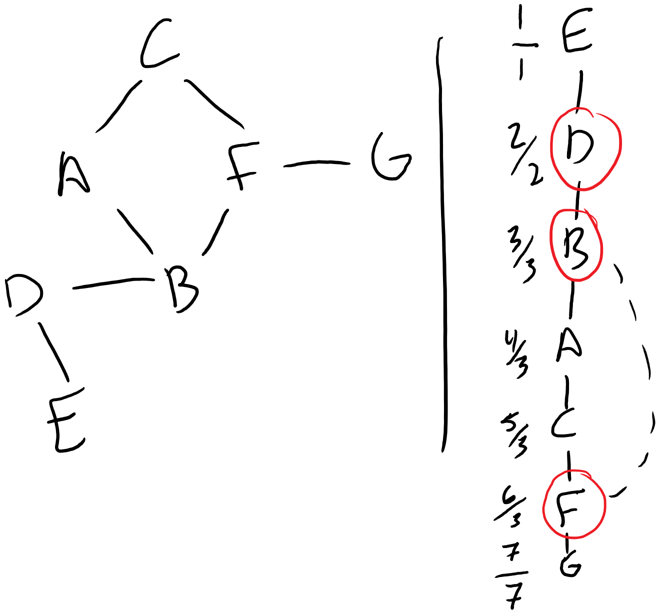
3. We represent this information as a fraction for convenience: ID / LOW VALUE

N 4. A node is an articulation point:
   a. At the root ***if and only if*** the root has two or more children
   b. At all other nodes when the node's child's LOW value >= its ID value

$O(N^2)$ ← too slow

- Can we reduce algorithmic efficiency to O(N)?
- By calculating LOW values first at leaf nodes and then progressively working up the tree, we can reduce this to O(N)

One more DFS articulation tree example


C
A
F
G
D
B
E
1/1 E
2/2 D
3/3 B
4/3 A
5/3 C
6/3 F
7/7 G