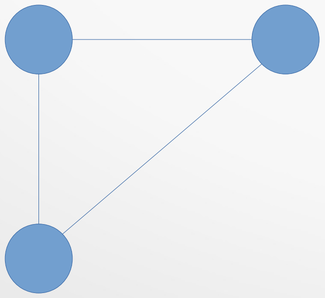
Biconnected Graphs

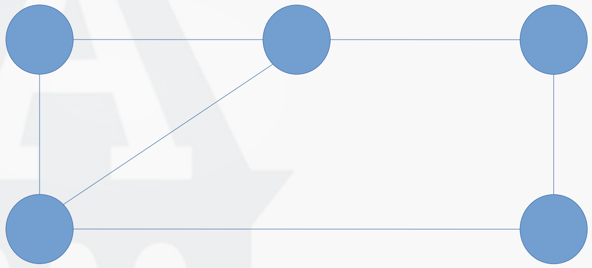
Biconnected Graphs

- An undirected graph is ***biconnected*** if there are no vertices whose removal disconnects the rest of the graph
- An ***articulation point*** is a node whose removal ***does*** disconnect the graph
- Identifying *back edges* helps in finding biconnected components

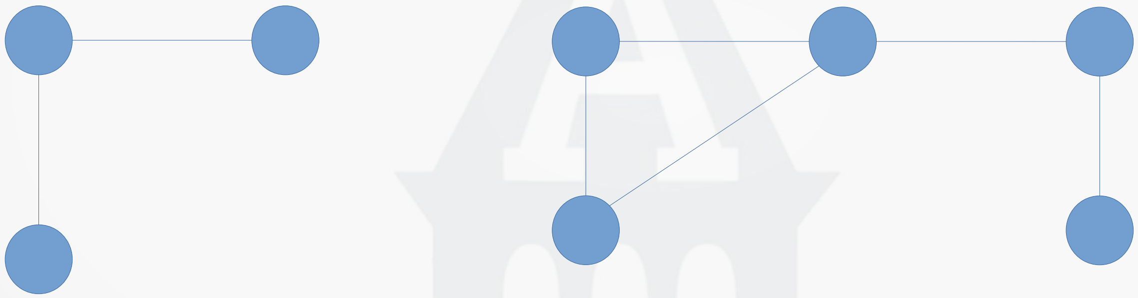
- This is important for computer networks, for example
  - Being able to route messages to all machines
- Also important for traffic (road & air) routes
  - Being able to travel from any point to any other point as roads are closed

Biconnected Examples





Not Biconnected Examples



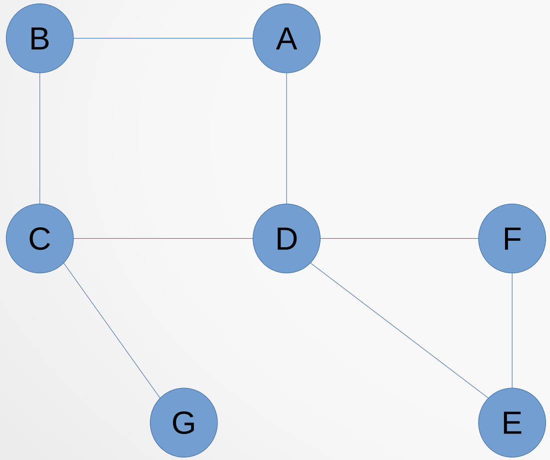
Biconnected Graphs - Algorithm

- Perform a depth first traversal, numbering the nodes as they are visited in pre-order
  - Call the numbers *Num(v)*
  - If a node has already been visited, the edge to it is a *back edge*
  - In this way, the undirected edges become directed by the order they are visited; these are *tree edges*
- For each node compute the lowest vertex that can be reached by ***zero or more tree edges***, followed by ***possibly one back edge***; this is called *Low(v)*

- Given *Num* and *Low* at each vertex, find articulation points as
  - The root is an articulation point if it has more than one child
  - Any other vertex v is an articulation point *iff* v has some child c, in the tree such that *Low(c)* >= *Num(v)*

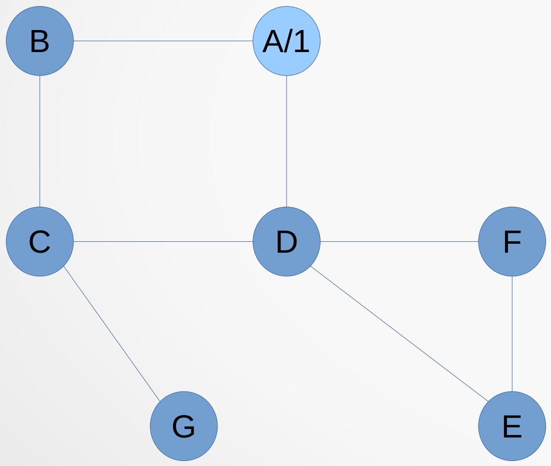
Biconnected Graphs – Simple Example

- Let’s do a simple example first



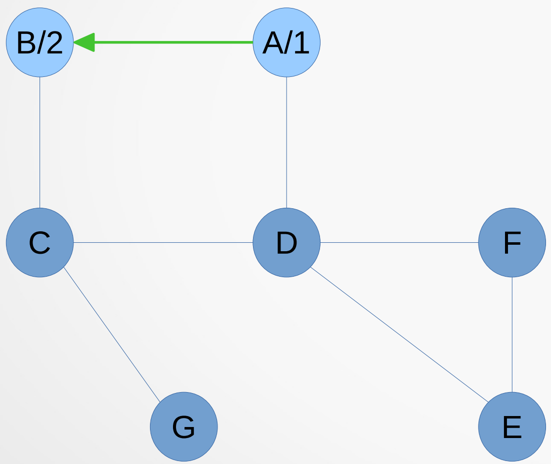
Biconnected Graphs – Simple Example

- Let's do a simple example first
  - Start at A, doing a depth first traversal, labeling as we go



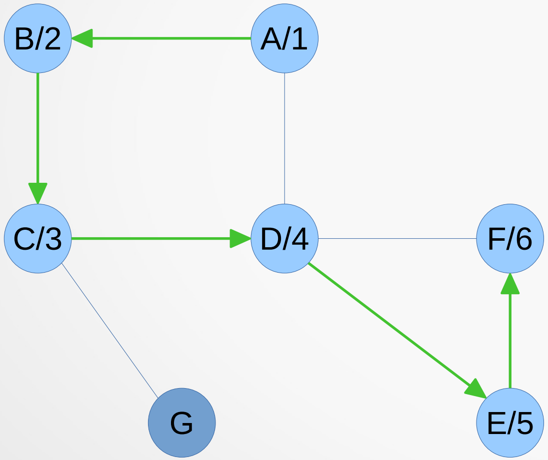
Biconnected Graphs – Simple Example

- Let's do a simple example first
  - Then to B



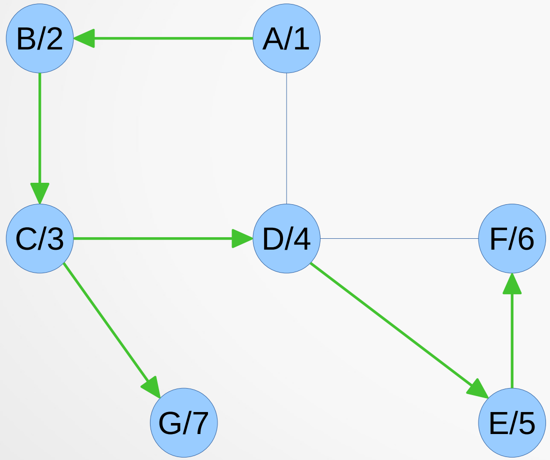
Biconnected Graphs – Simple Example

- Let's do a simple example first
  - Then to C, D, E, F



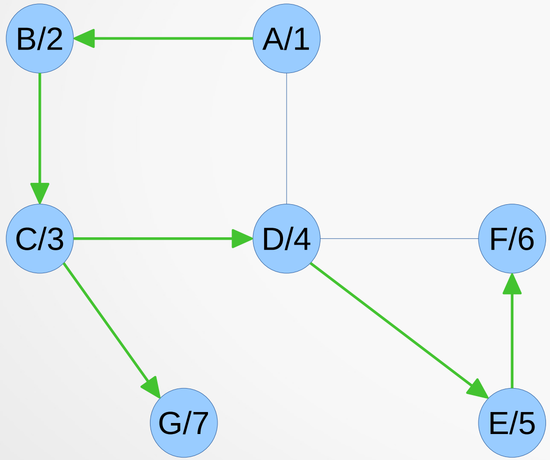
Biconnected Graphs – Simple Example

- Let’s do a simple example first
  - Back up to C, then to G

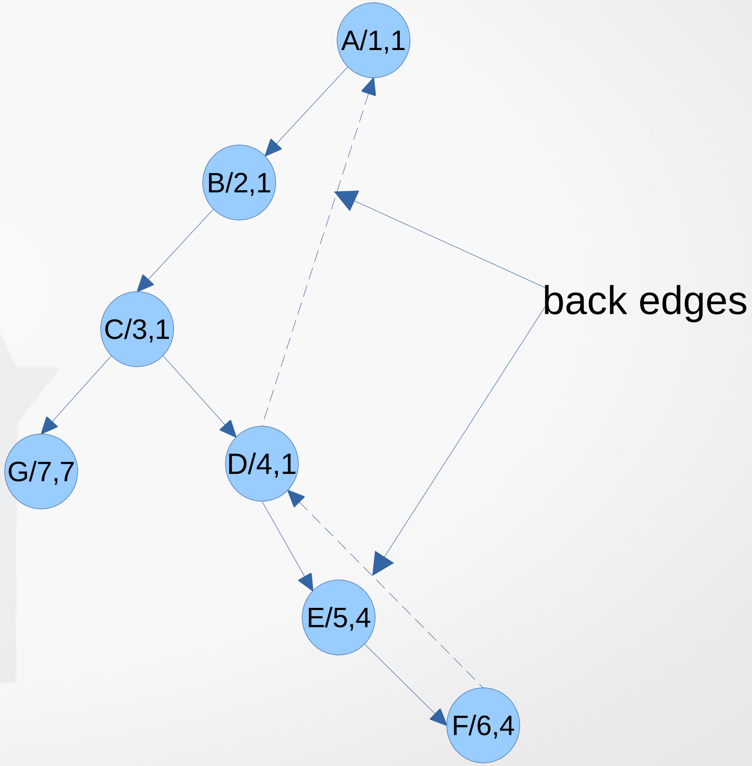


Biconnected Graphs – Simple Example

- Let's do a simple example first
  - Rearrange graph, organizing as a tree



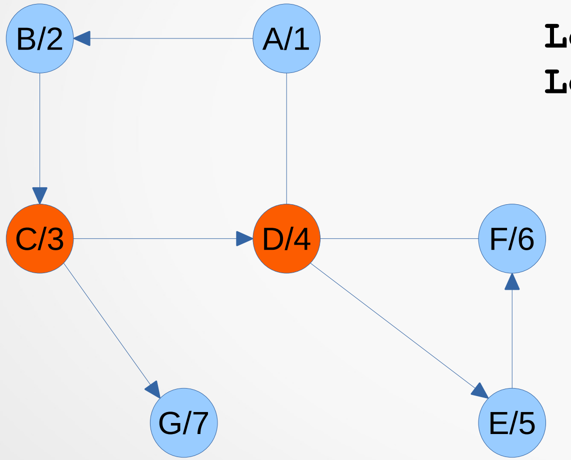
Node Labeling: Name/Num(v),Low(v)



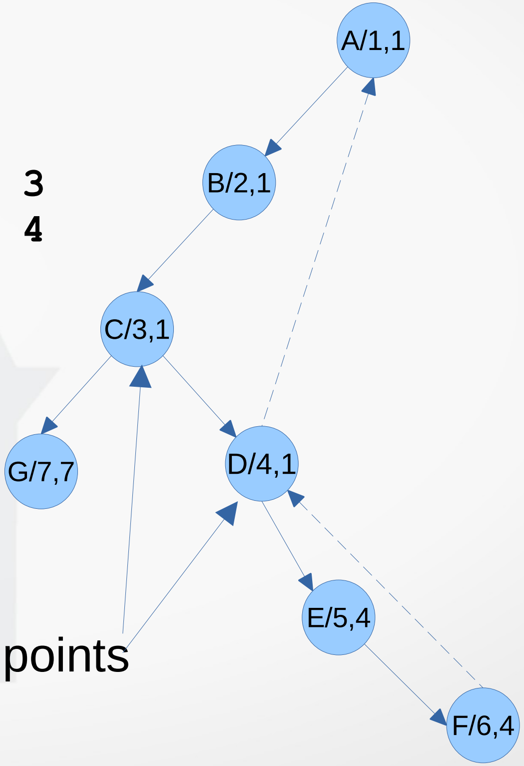
Low(v): smallest node it can reach using only one back edge

Biconnected Graphs – Simple Example

- Let’s do a simple example first
  - Find articulation points



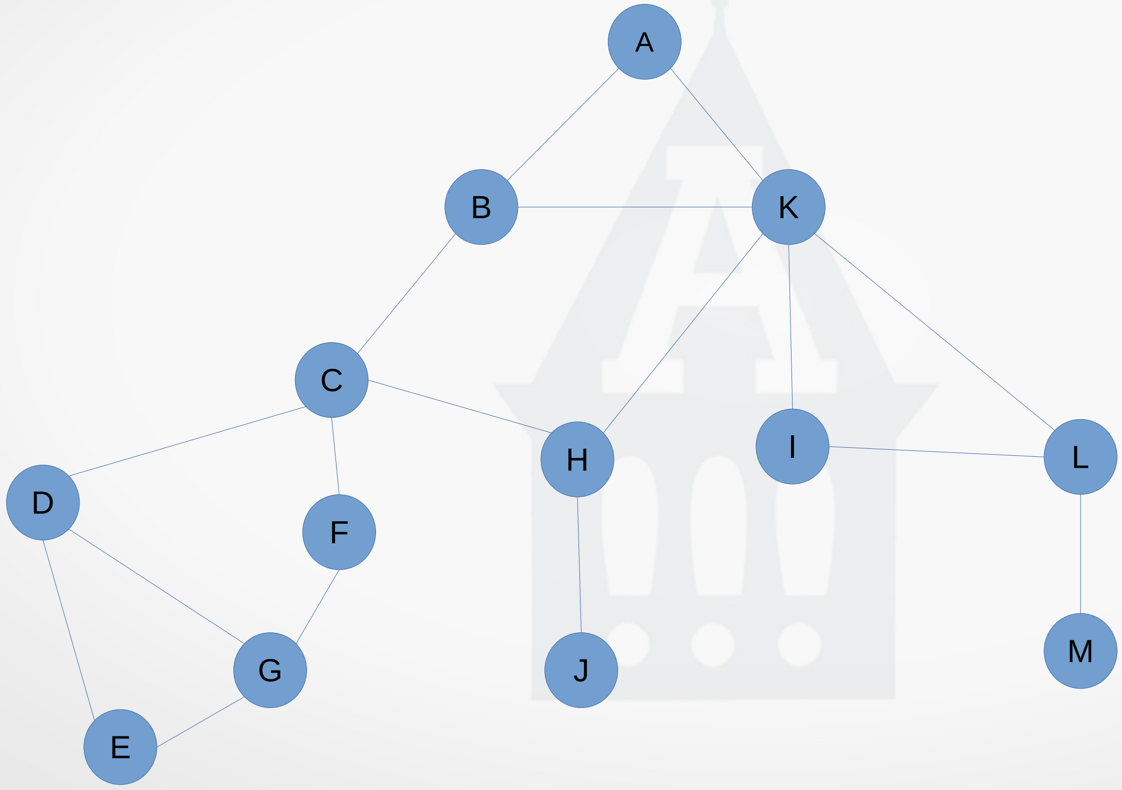

```
Low(G) 7 >= Num(C) 3
Low(E) 4 >= Num(D) 4
```



articulation points

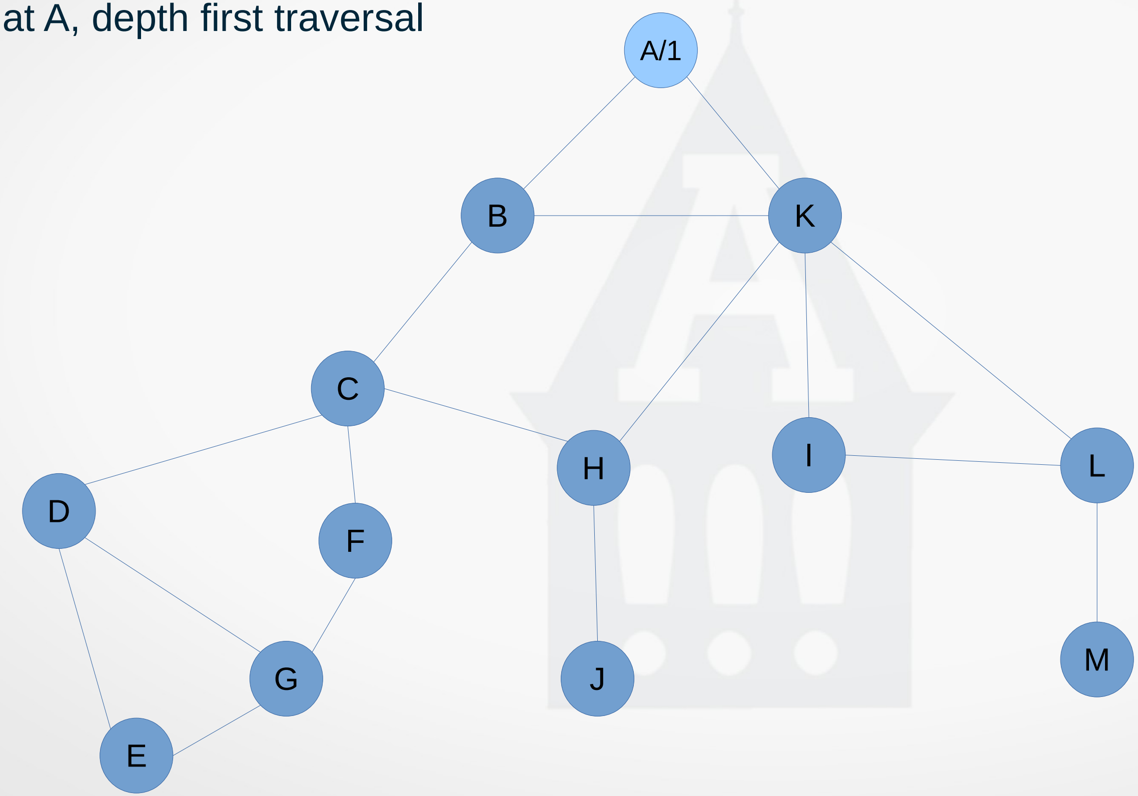
Low(v): smallest node it can reach using only one back edge

Biconnected Graphs – Example 2



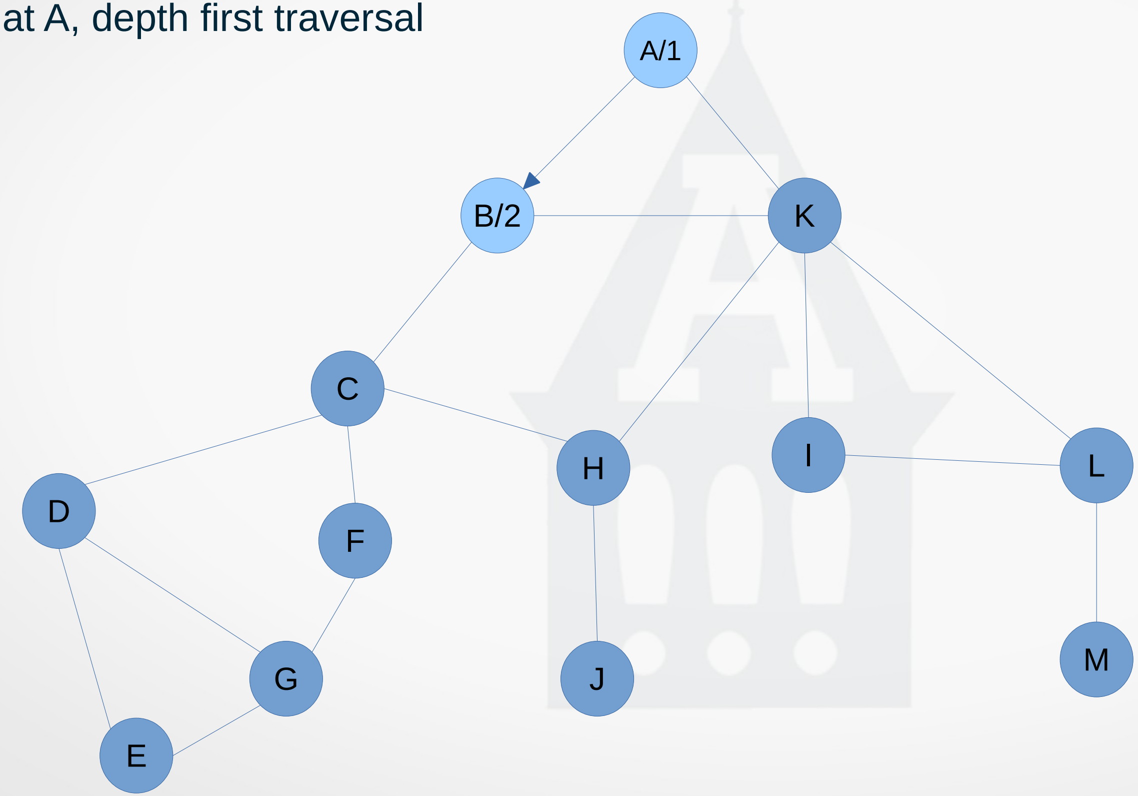
Biconnected Graphs – Example 2

- Start at A, depth first traversal



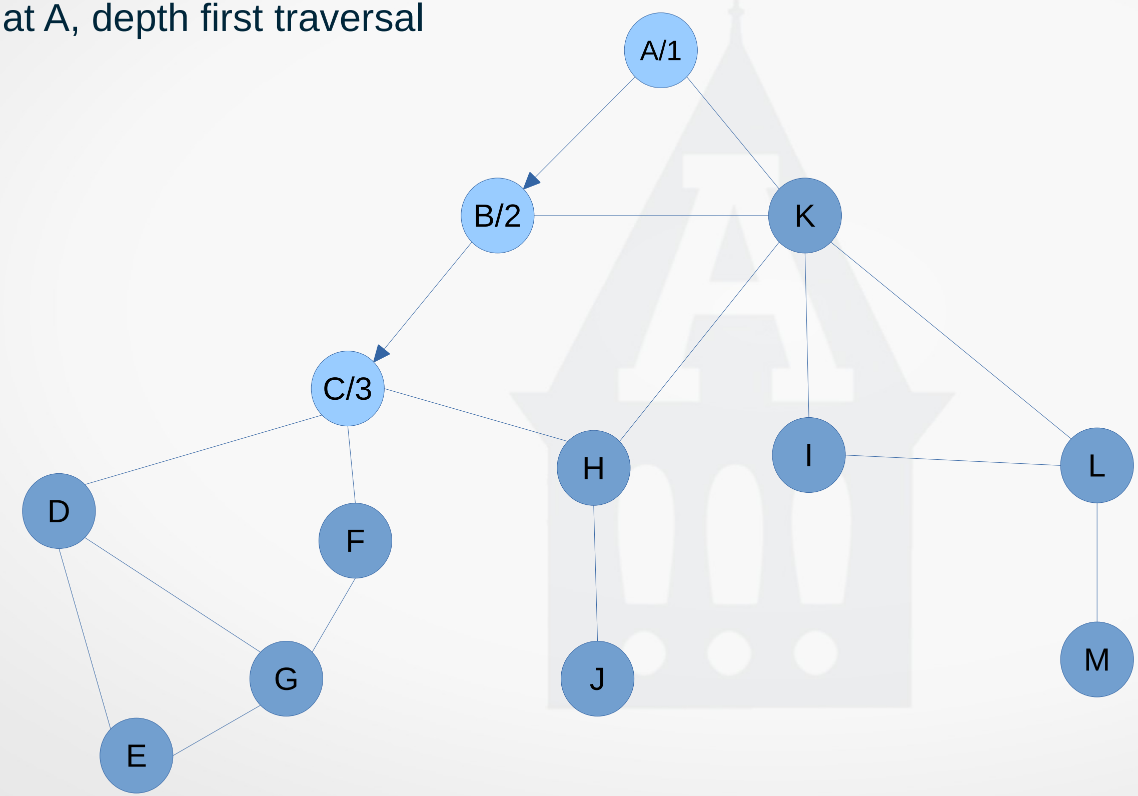
Biconnected Graphs – Example 2

- Start at A, depth first traversal



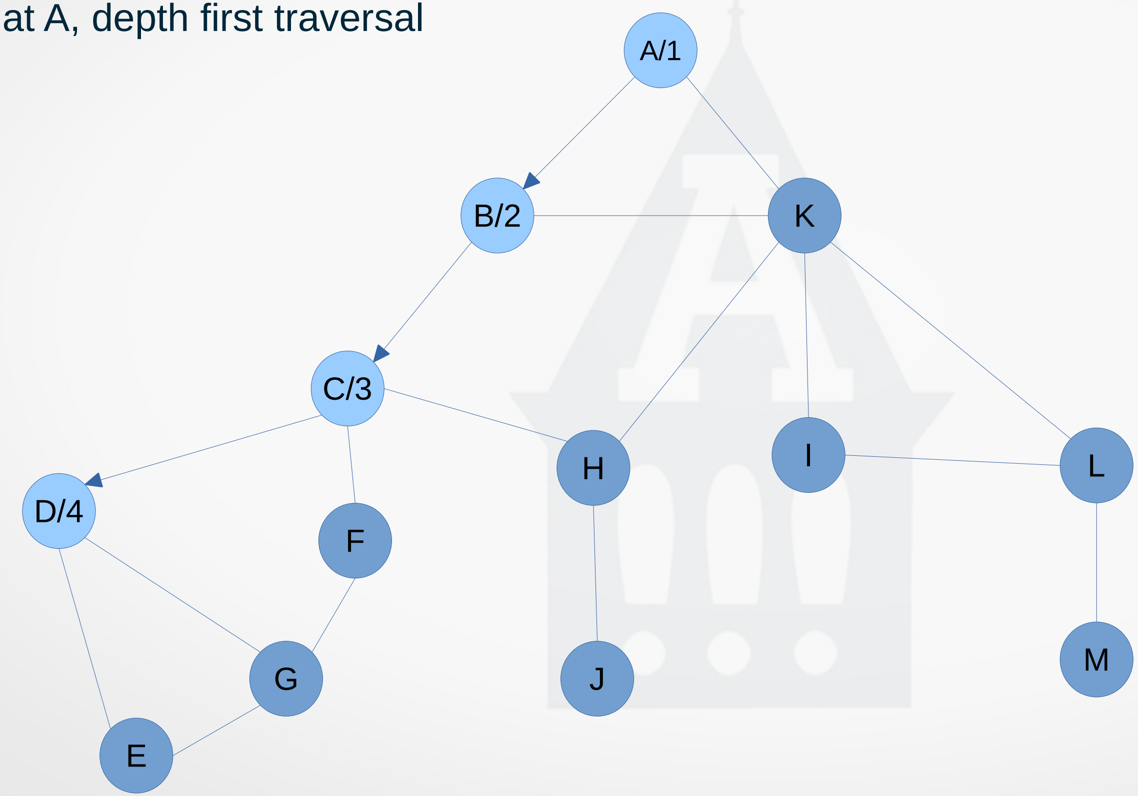
Biconnected Graphs – Example 2

- Start at A, depth first traversal



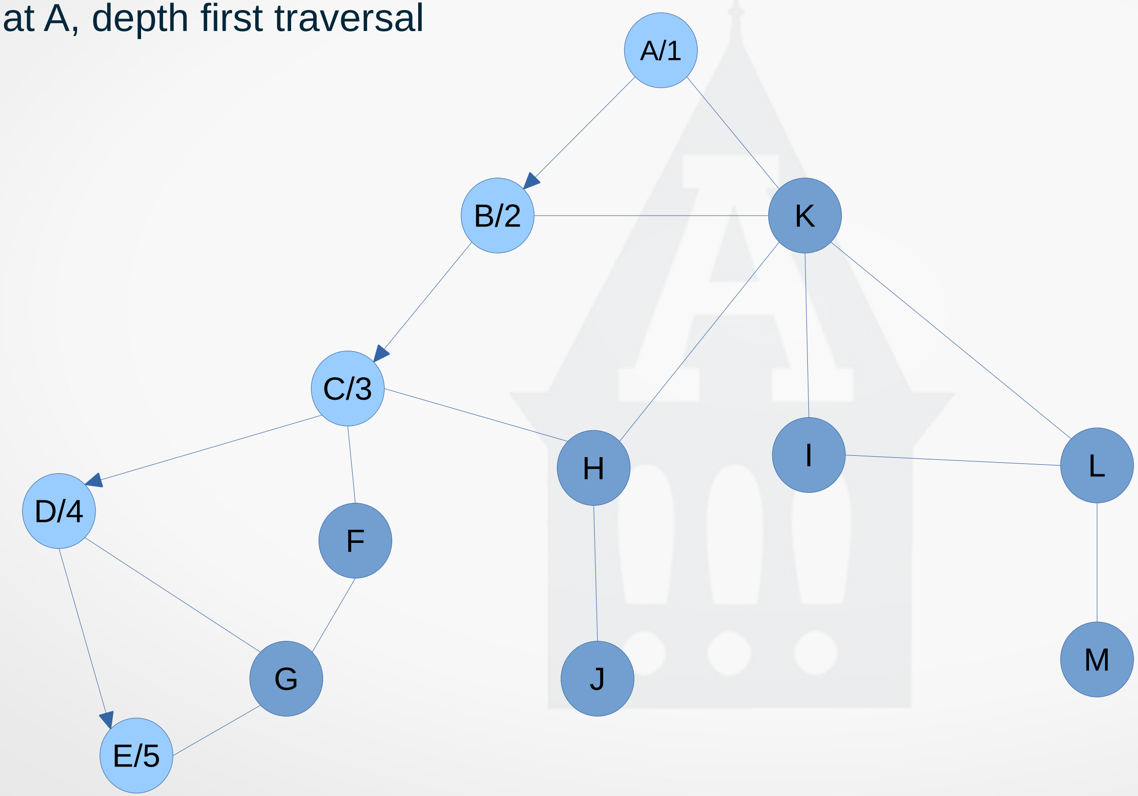
Biconnected Graphs – Example 2

- Start at A, depth first traversal



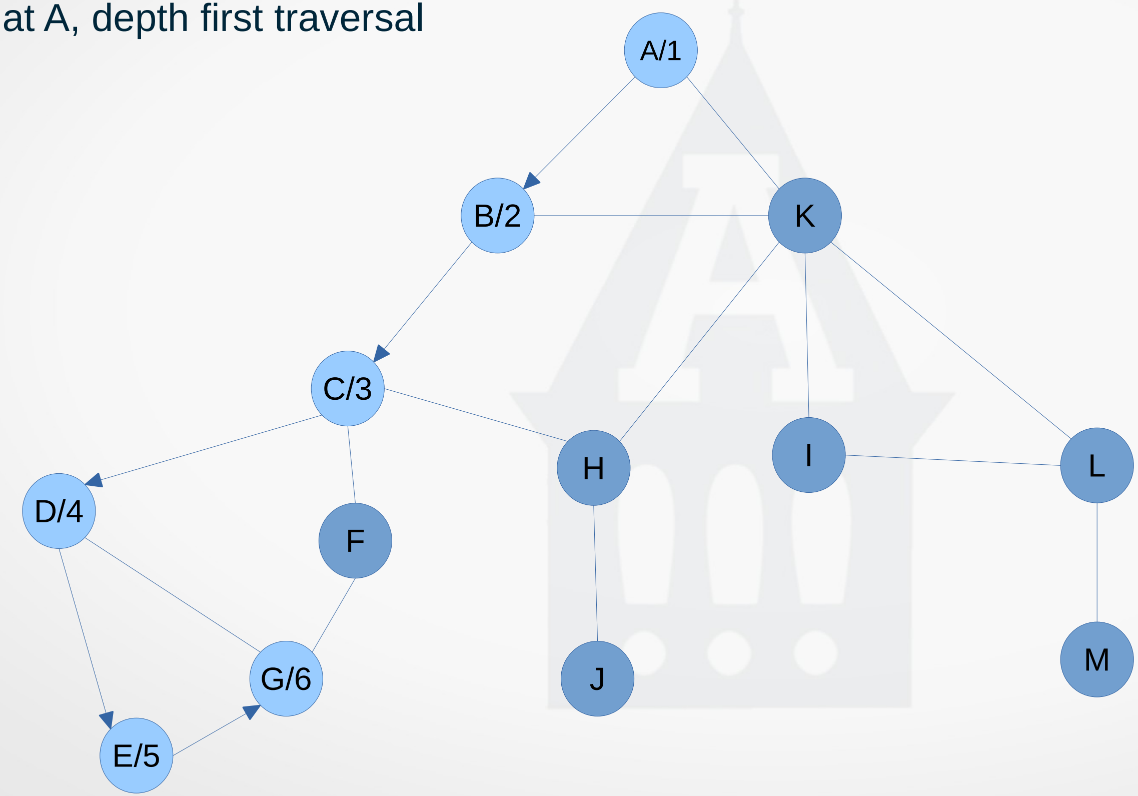
Biconnected Graphs – Example 2

- Start at A, depth first traversal



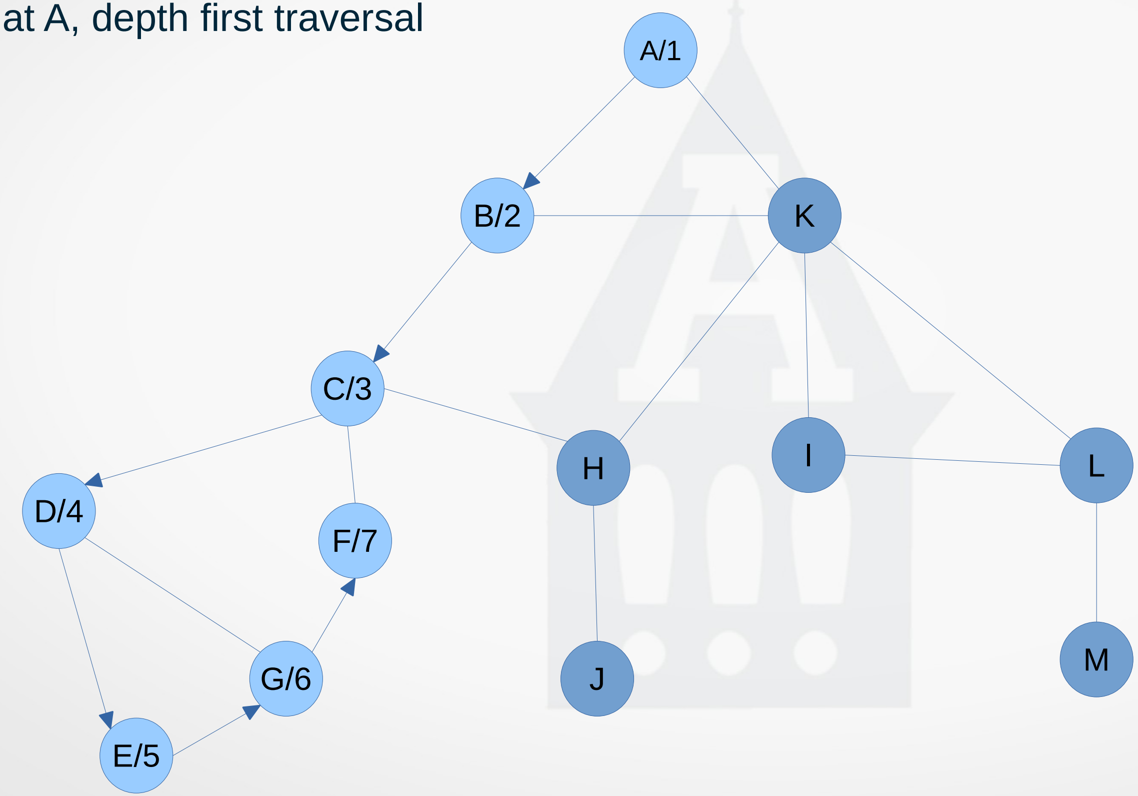
Biconnected Graphs – Example 2

- Start at A, depth first traversal



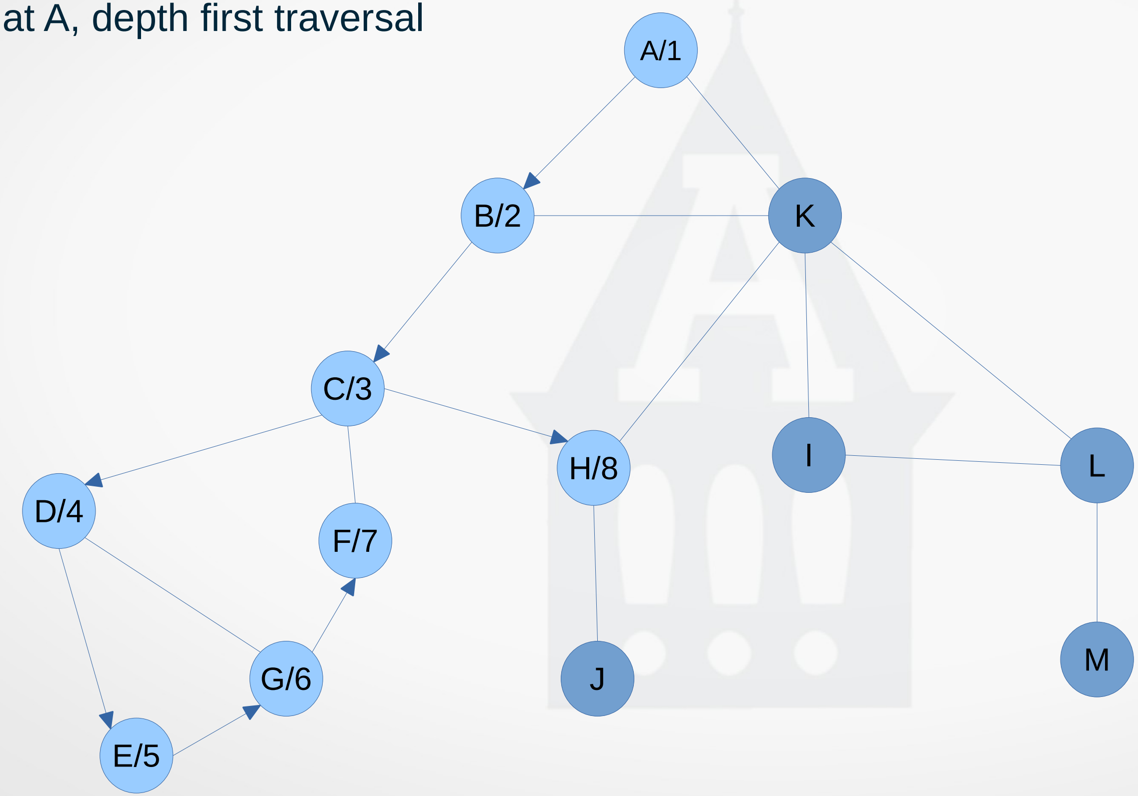
Biconnected Graphs – Example 2

- Start at A, depth first traversal



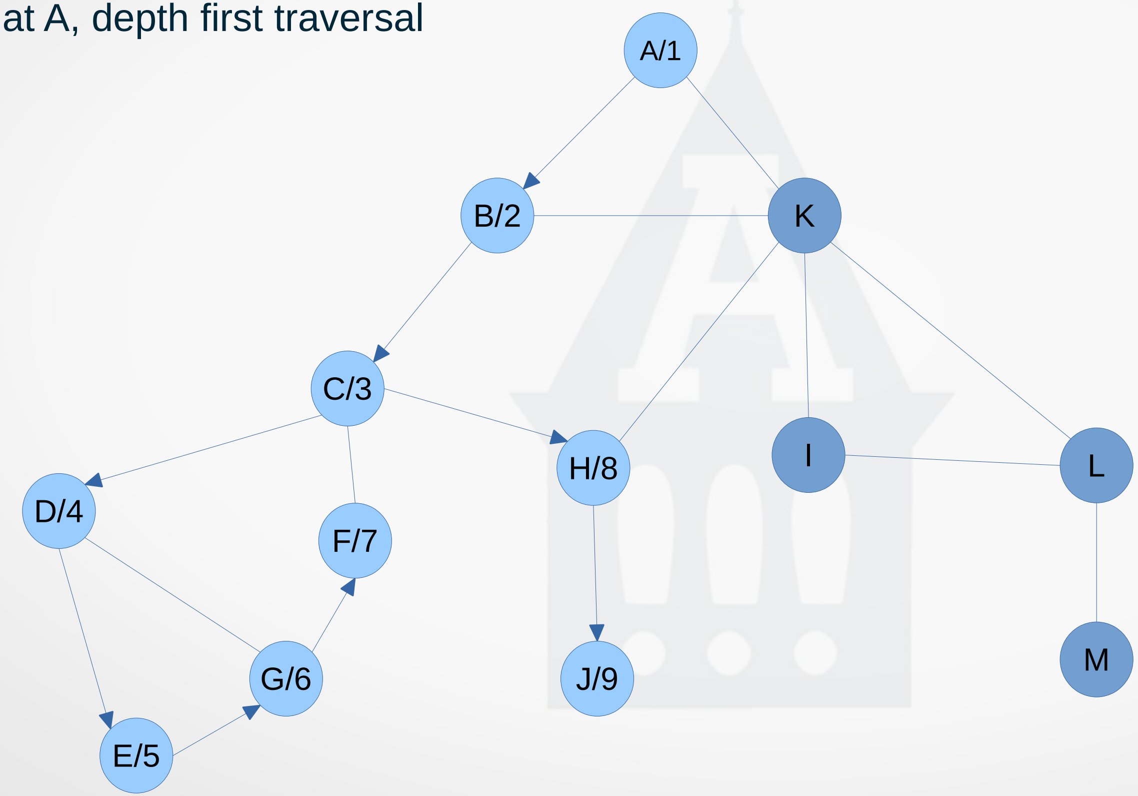
Biconnected Graphs – Example 2

- Start at A, depth first traversal



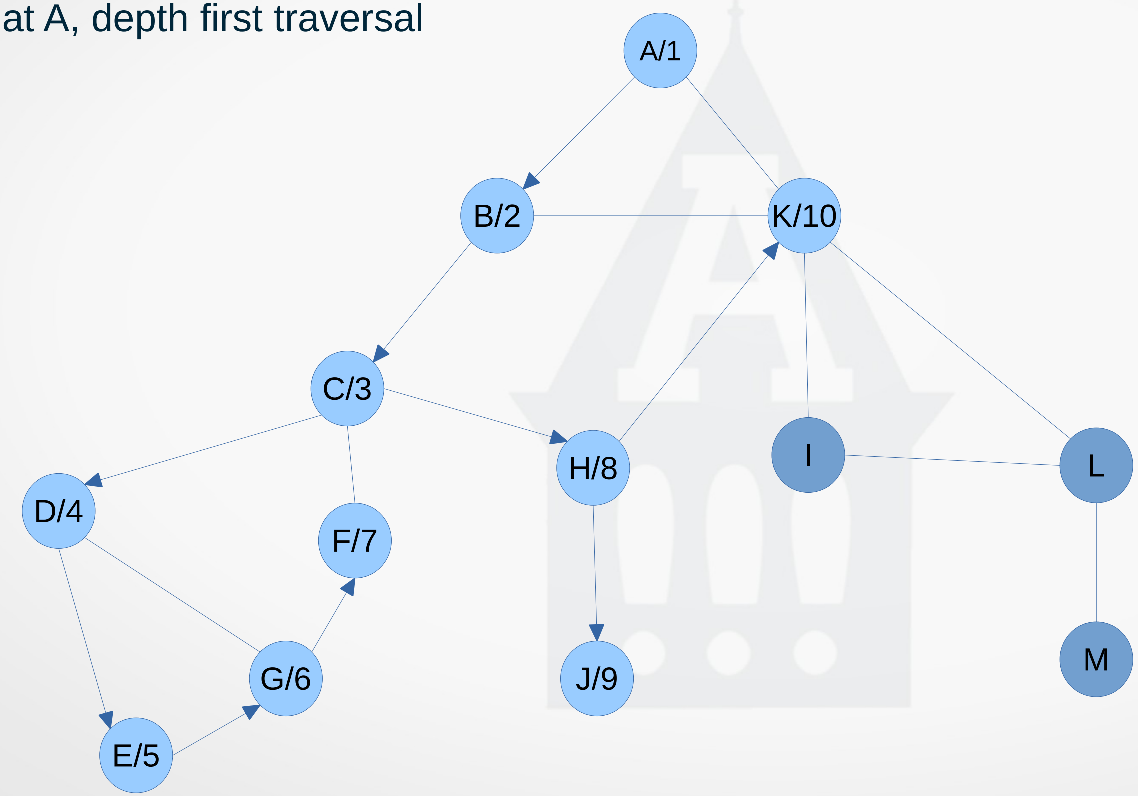
Biconnected Graphs – Example 2

- Start at A, depth first traversal



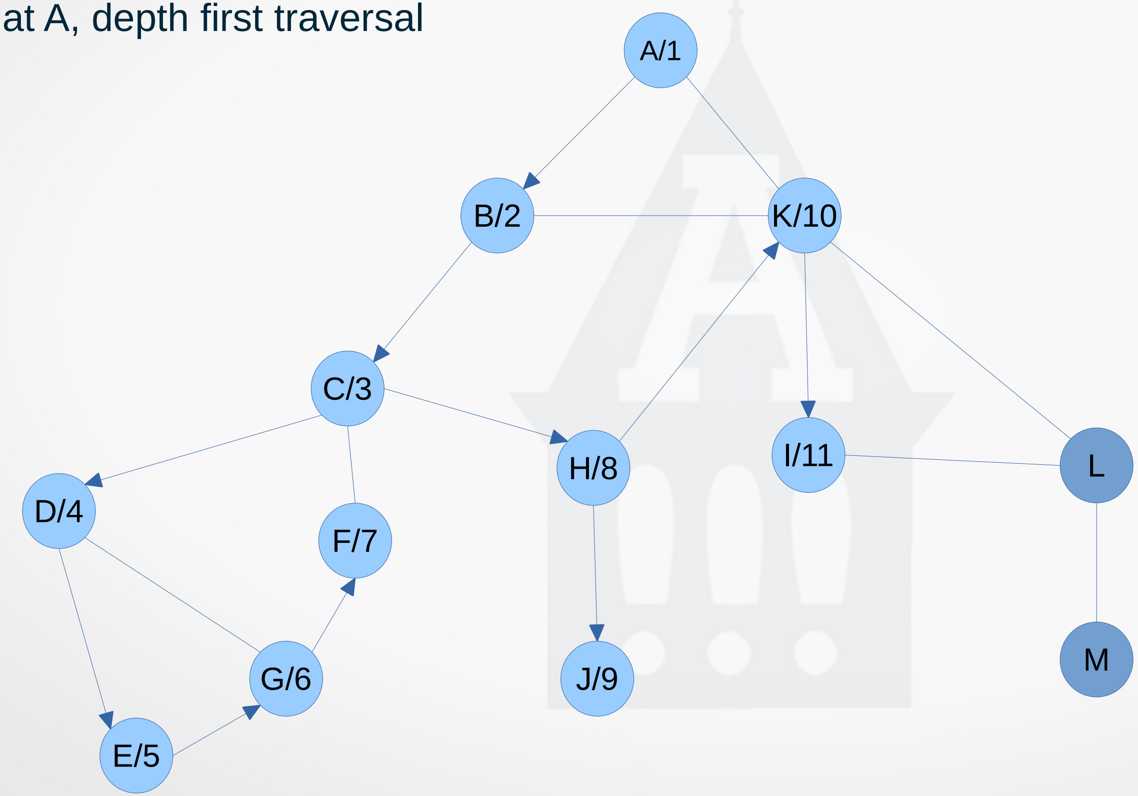
Biconnected Graphs – Example 2

- Start at A, depth first traversal



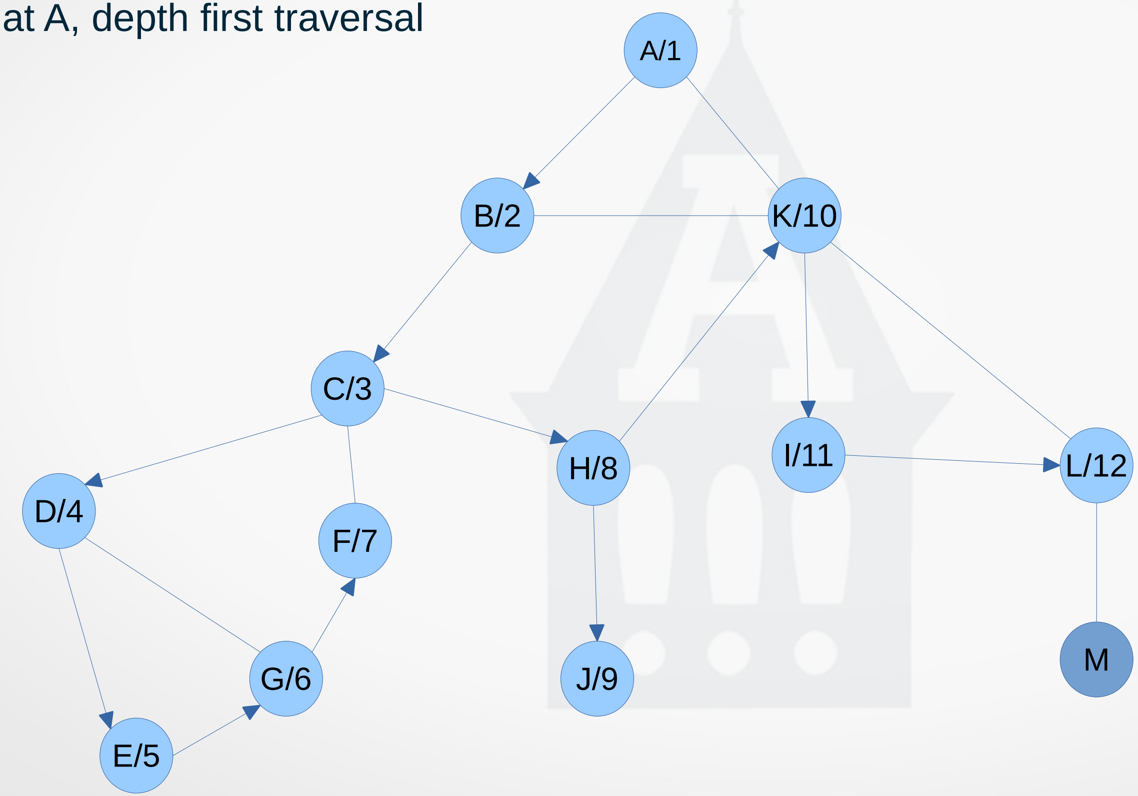
Biconnected Graphs – Example 2

- Start at A, depth first traversal



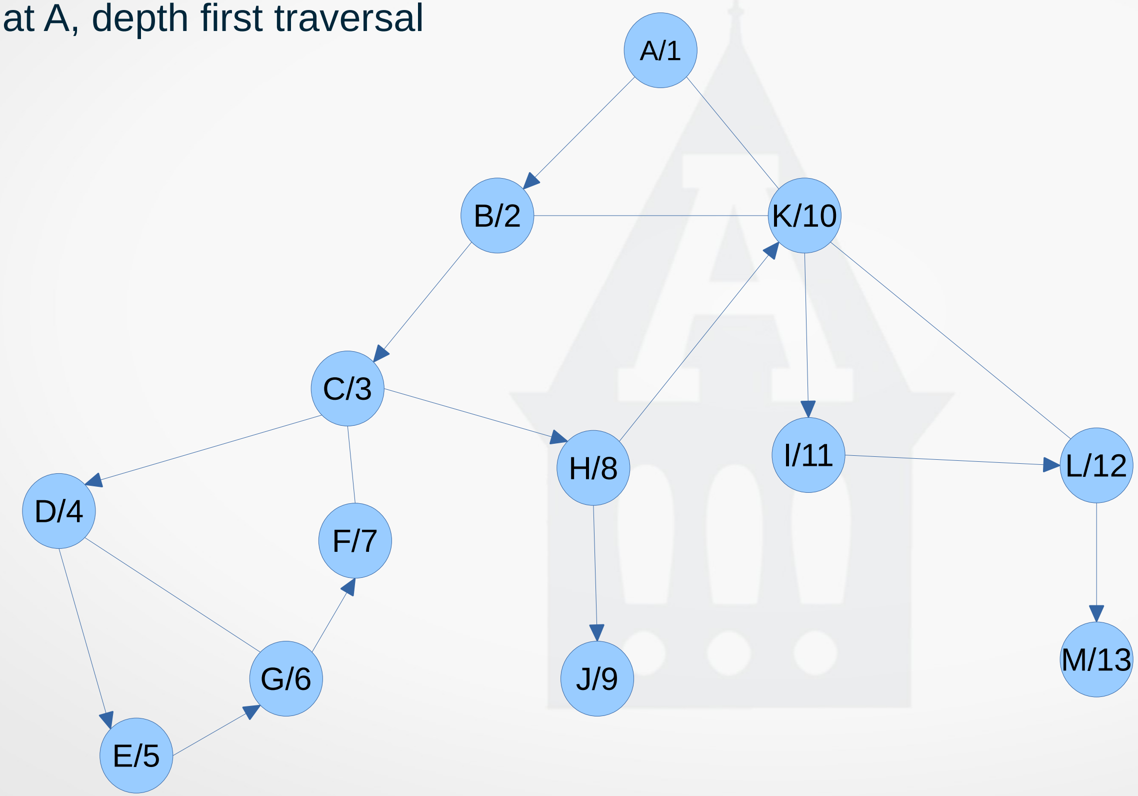
Biconnected Graphs – Example 2

- Start at A, depth first traversal



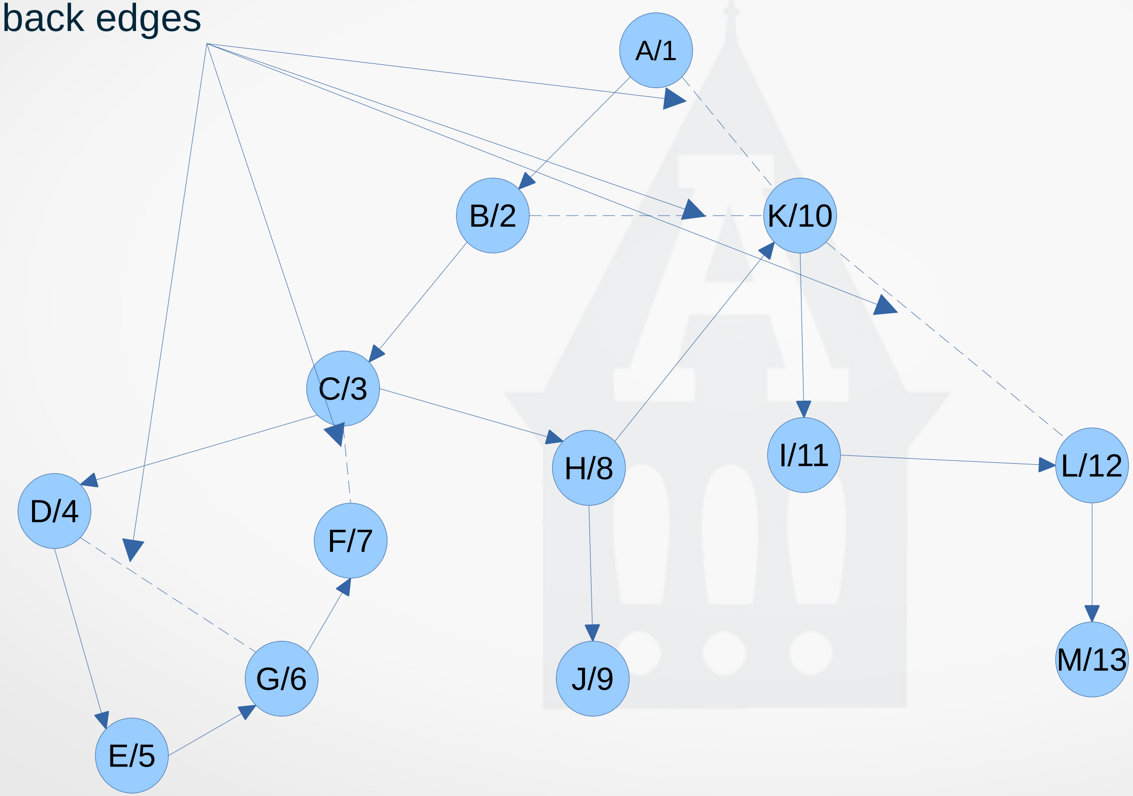
Biconnected Graphs – Example 2

- Start at A, depth first traversal



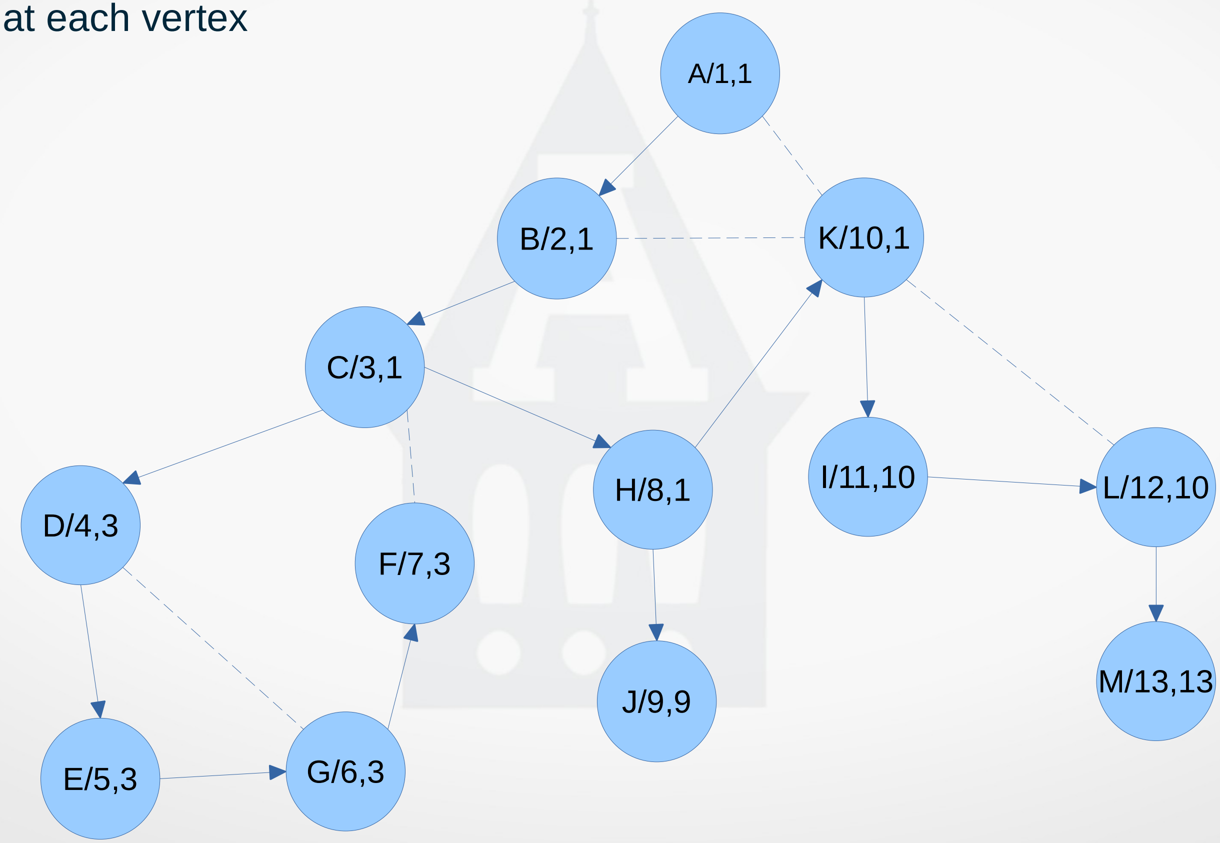
Biconnected Graphs – Example 2

- Mark back edges



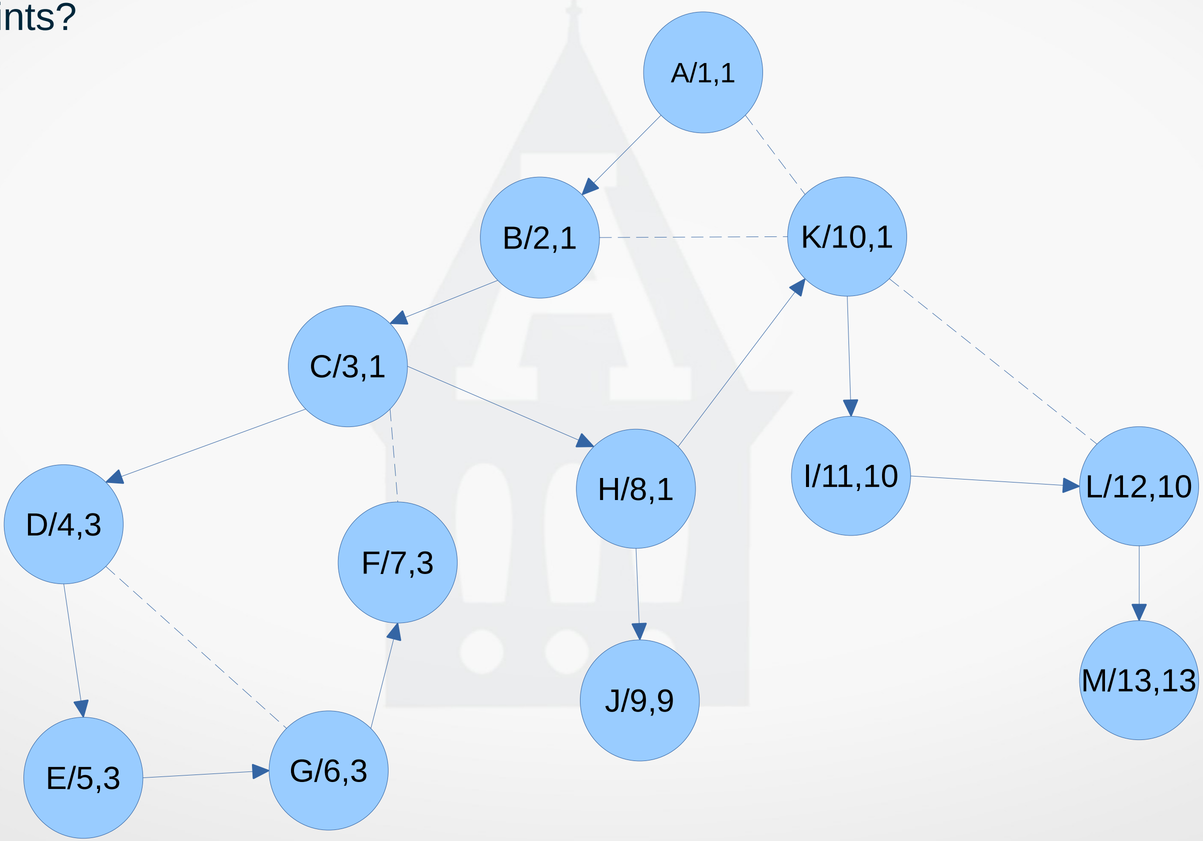
Biconnected Graphs – Example 2

- Compute low at each vertex



Biconnected Graphs – Example 2

- Articulation points?



Biconnected Graphs – Example 2

- Articulation points?
  - `low(D)   3 >= num(C)   3`
  - `low(J)   9 >= num(H)   8`
  - `low(I)  10 >= num(K)   1`
  - `low(M)  13 >= num(L)  10`