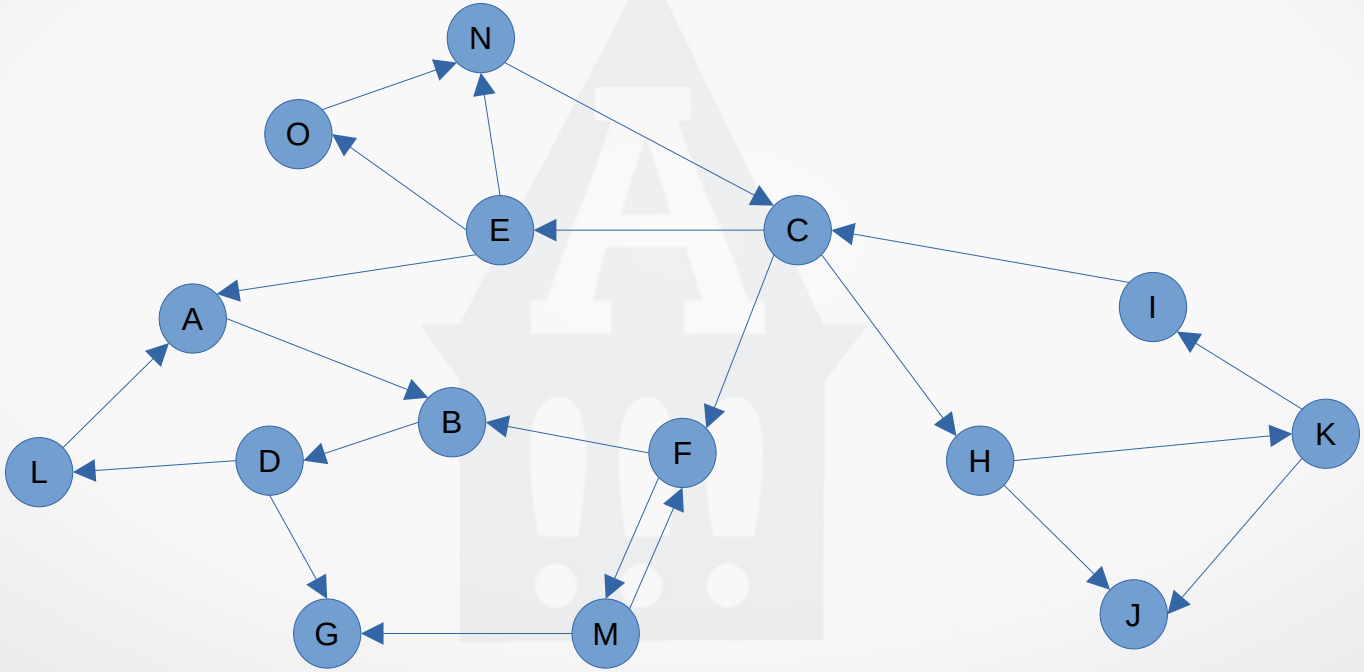
Strongly Connected Components

Finding Strongly Connected Components

- Definition: Every pair of vertices are reachable from each other
- Perform a depth first traversal, numbering the nodes in post-order
  - Mark as visited during the traversal
  - Wait to order until coming out of the recursion (post-order)

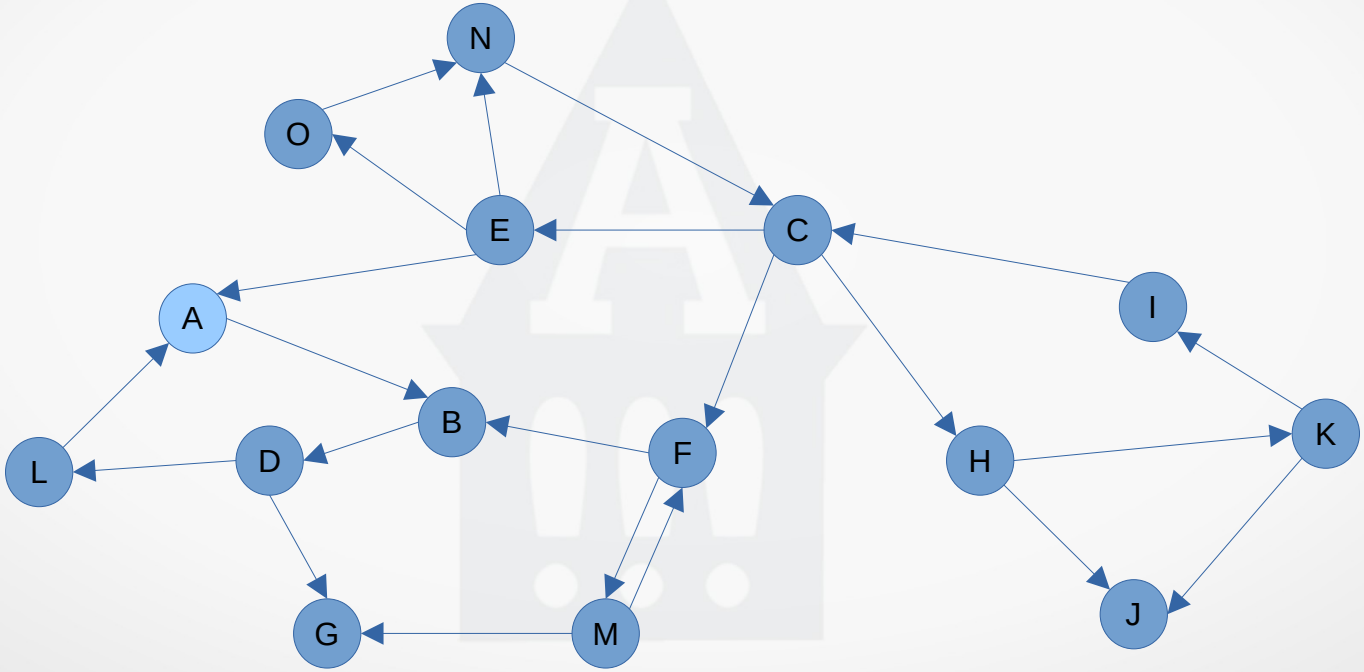
```
numberByDFS(node)
    node.visited = true
    for each successor of node
        if successor is unvisited
            numberByDFS(successor)
    node.number = nextNumber++
```

Finding Strongly Connected Components



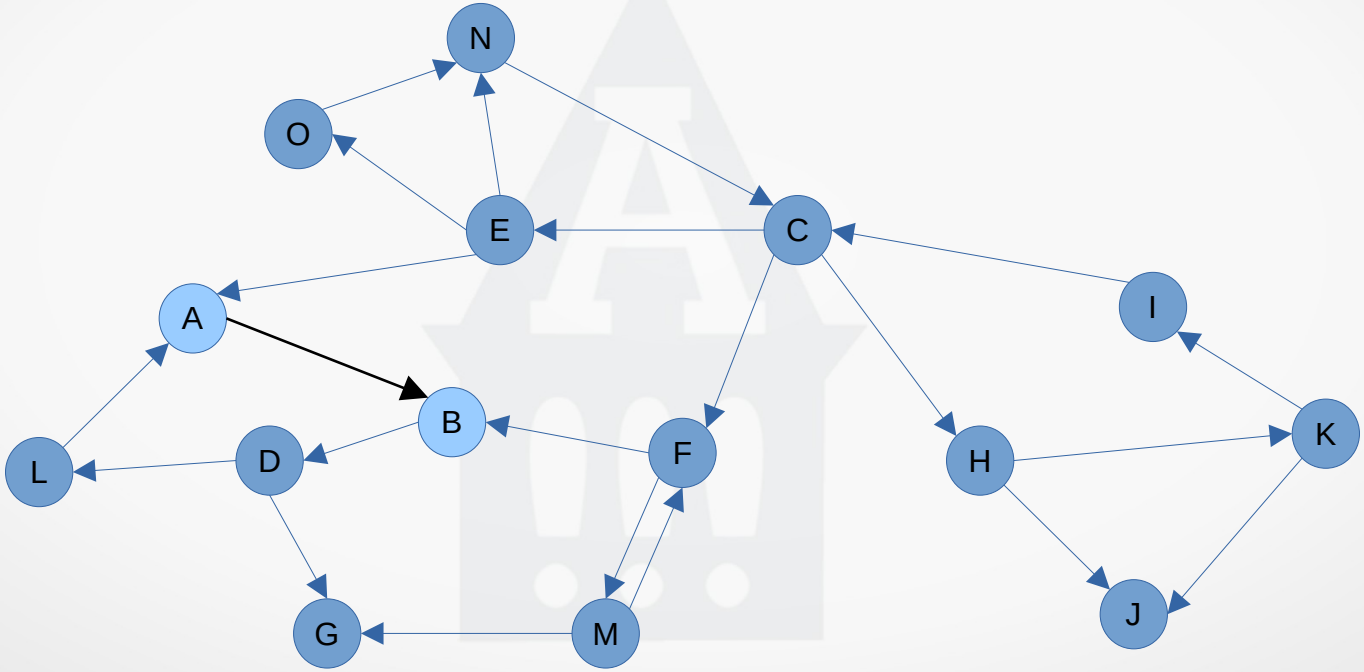
Finding Strongly Connected Components

- Let's start at A



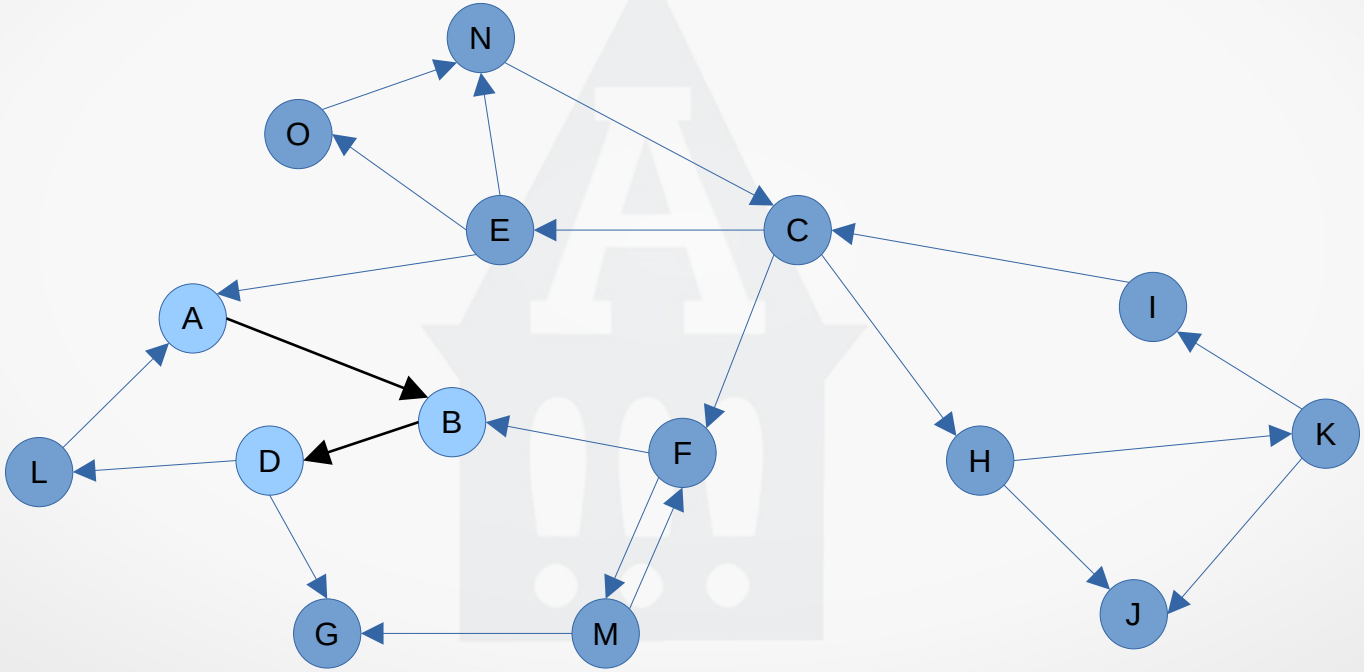
Finding Strongly Connected Components

- Let's start at A, then B



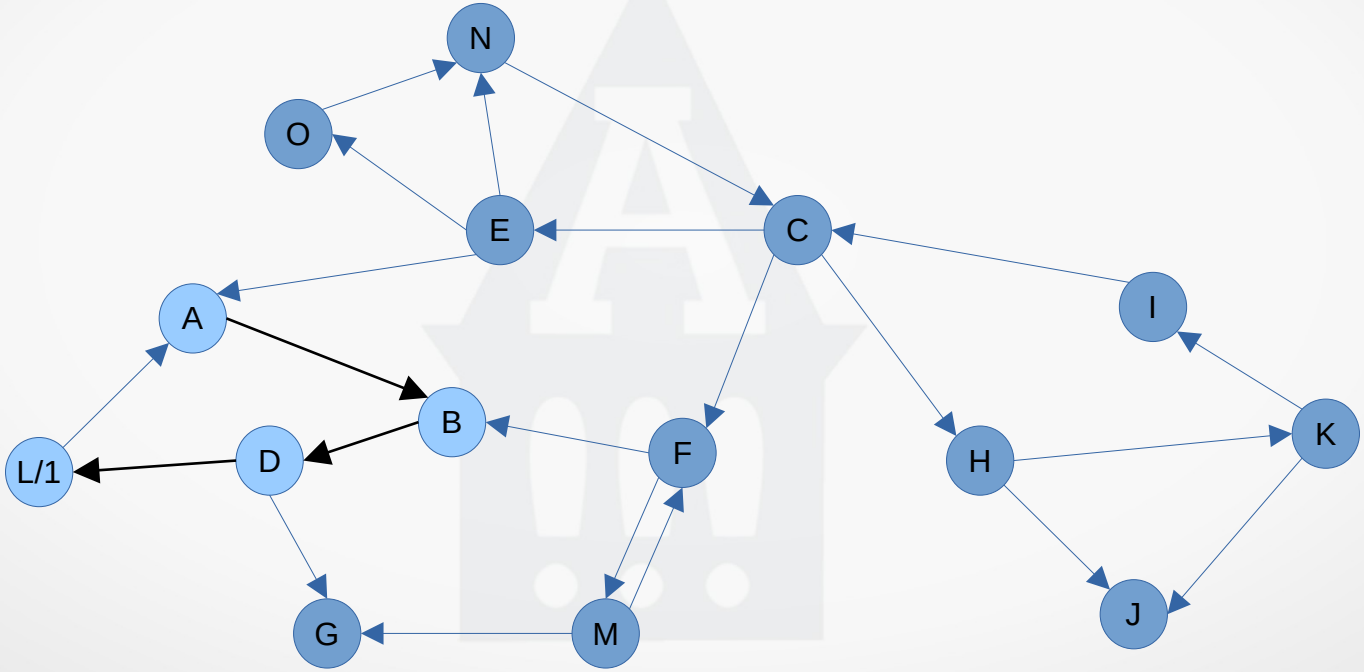
Finding Strongly Connected Components

- Let's start at A, then D



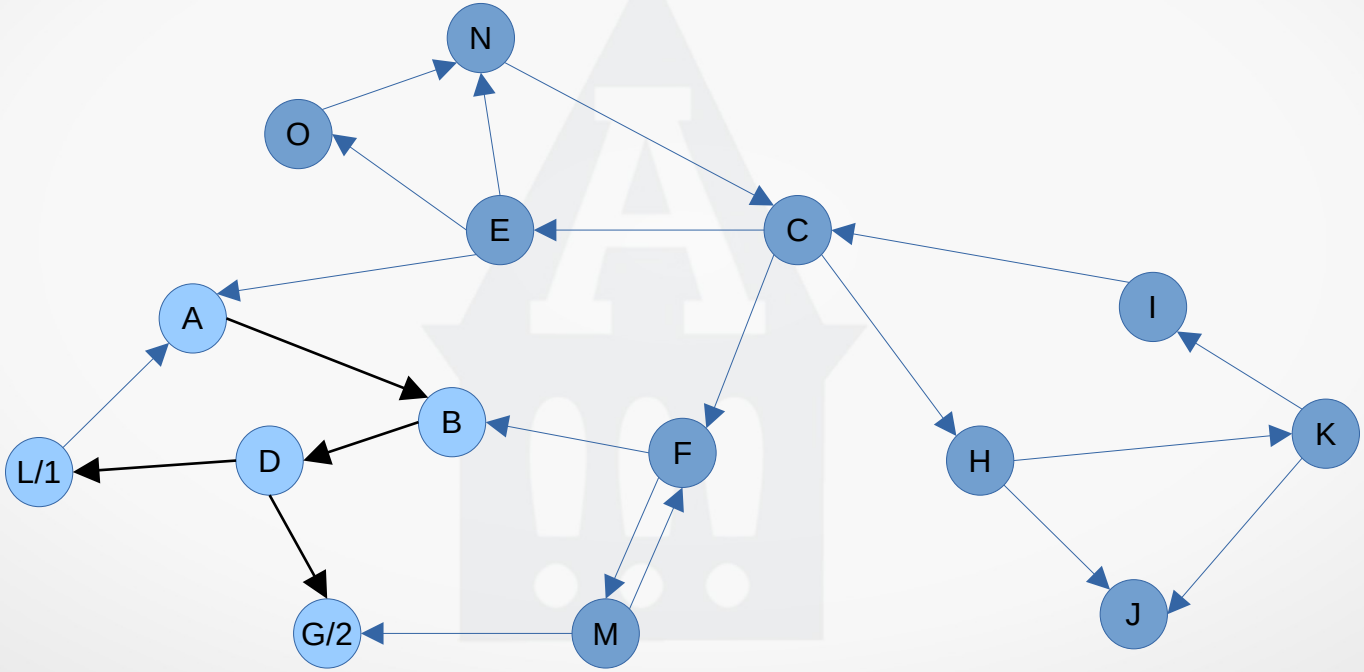
Finding Strongly Connected Components

- Let’s start at A, then D, then L; but have already visited A, number L



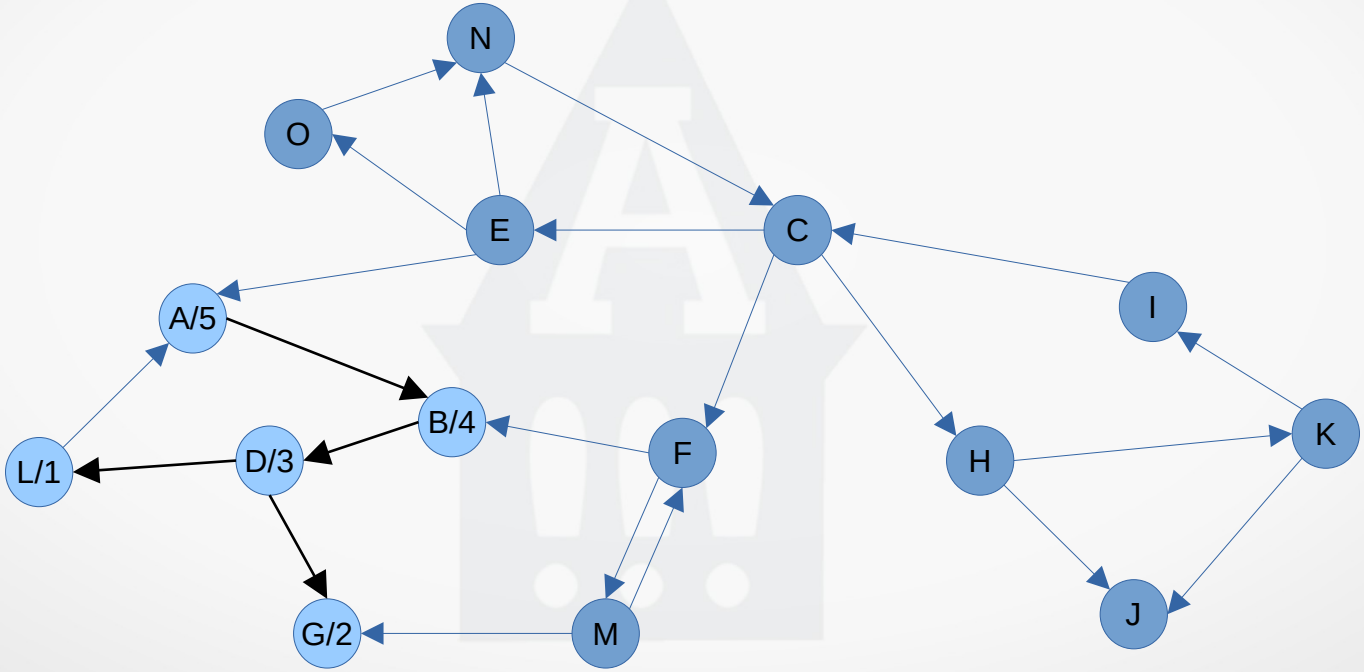
Finding Strongly Connected Components

- Let’s start at A, then D, then L, number L, then G, no where else to go, number G



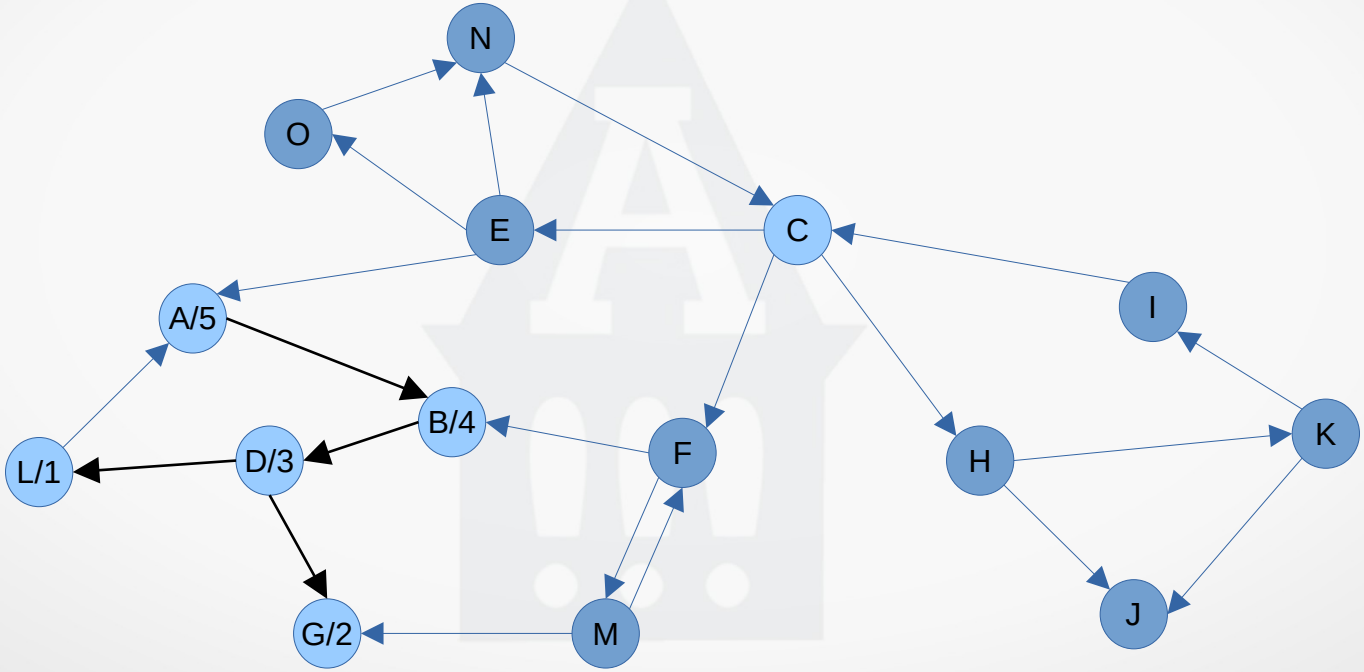
Finding Strongly Connected Components

- Number the rest on the way back out



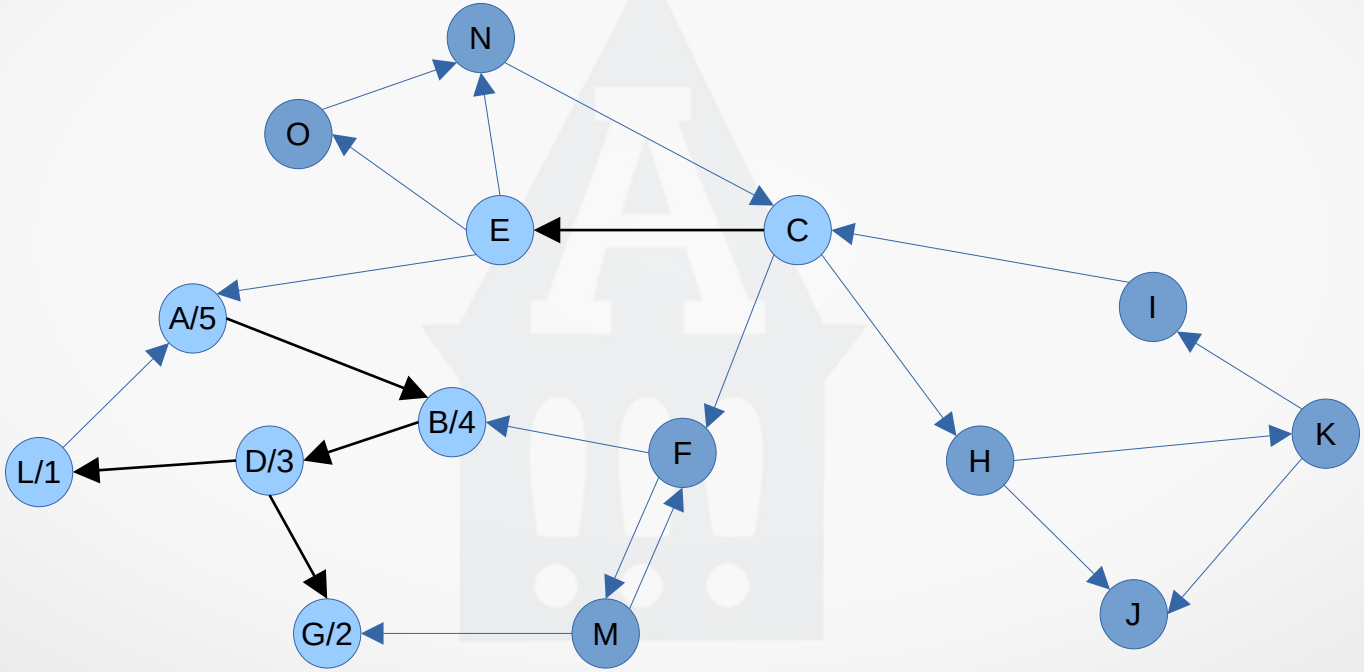
Finding Strongly Connected Components

- Start over again with C; following alphabetical order (this doesn't actually matter)



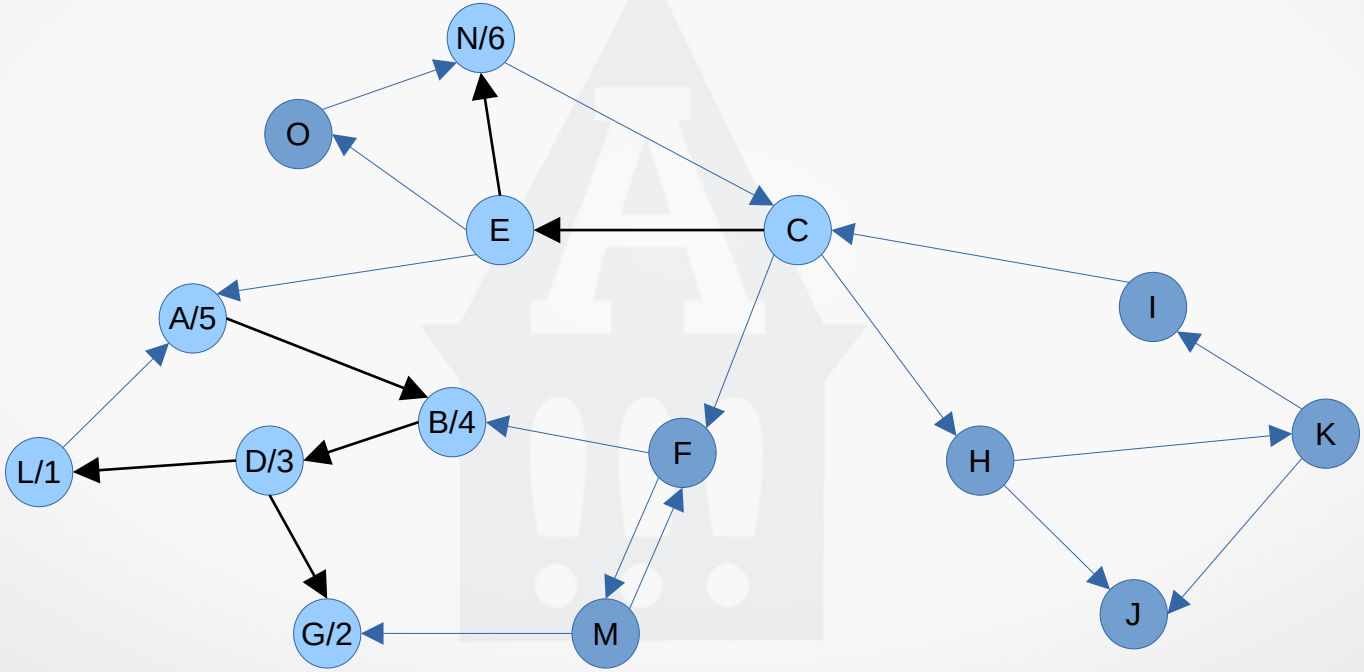
Finding Strongly Connected Components

- Start over again with C, then E



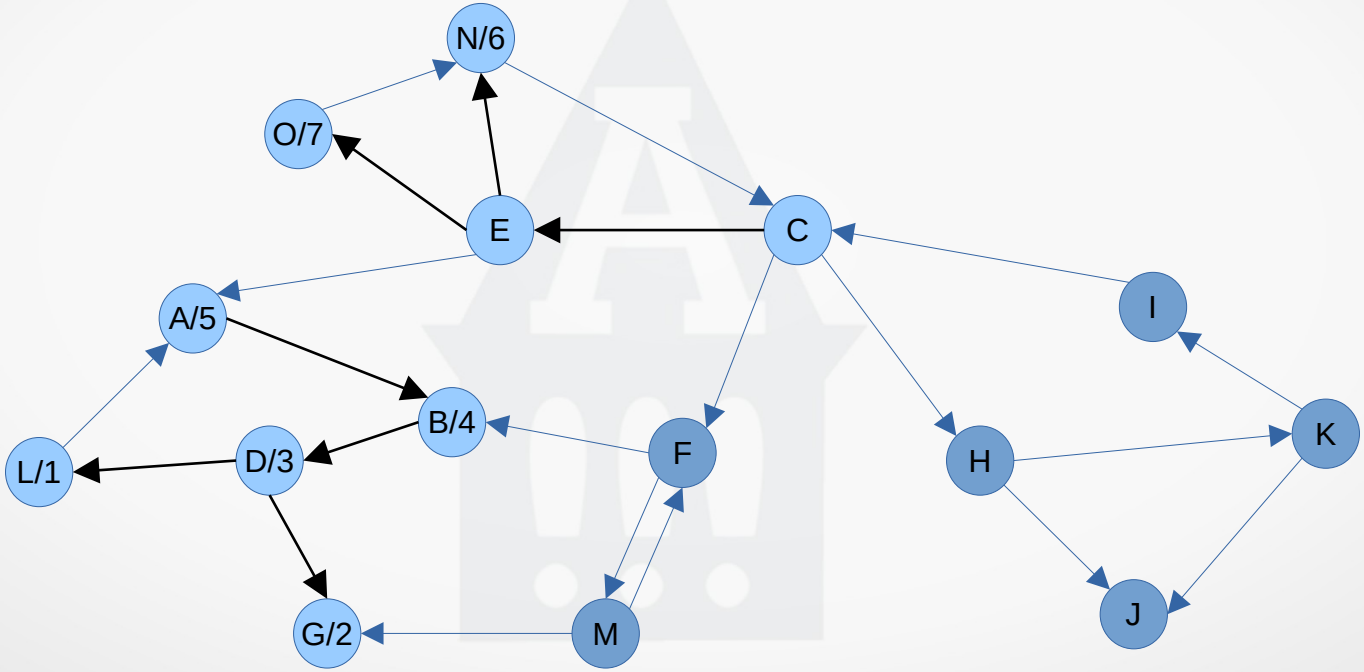
Finding Strongly Connected Components

- Start over again with C, then E, then N; number it with 6



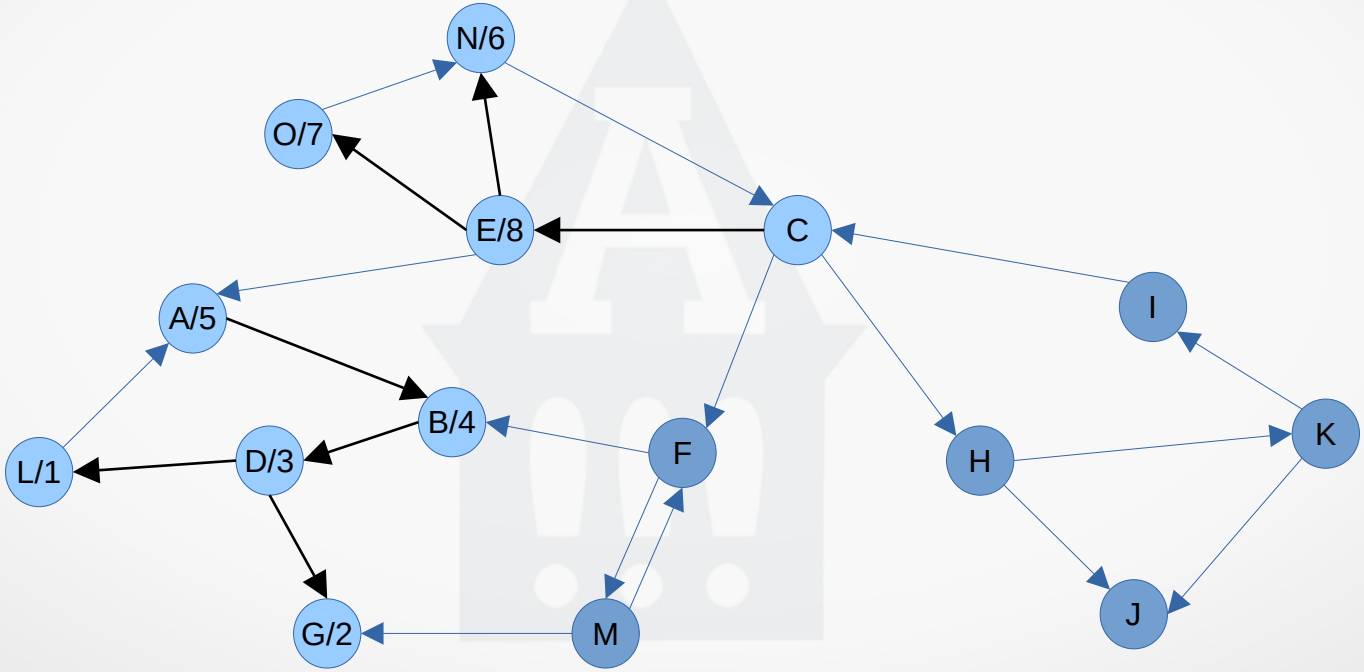
Finding Strongly Connected Components

- Start over again with C, then E, then N, then O; number it with 7



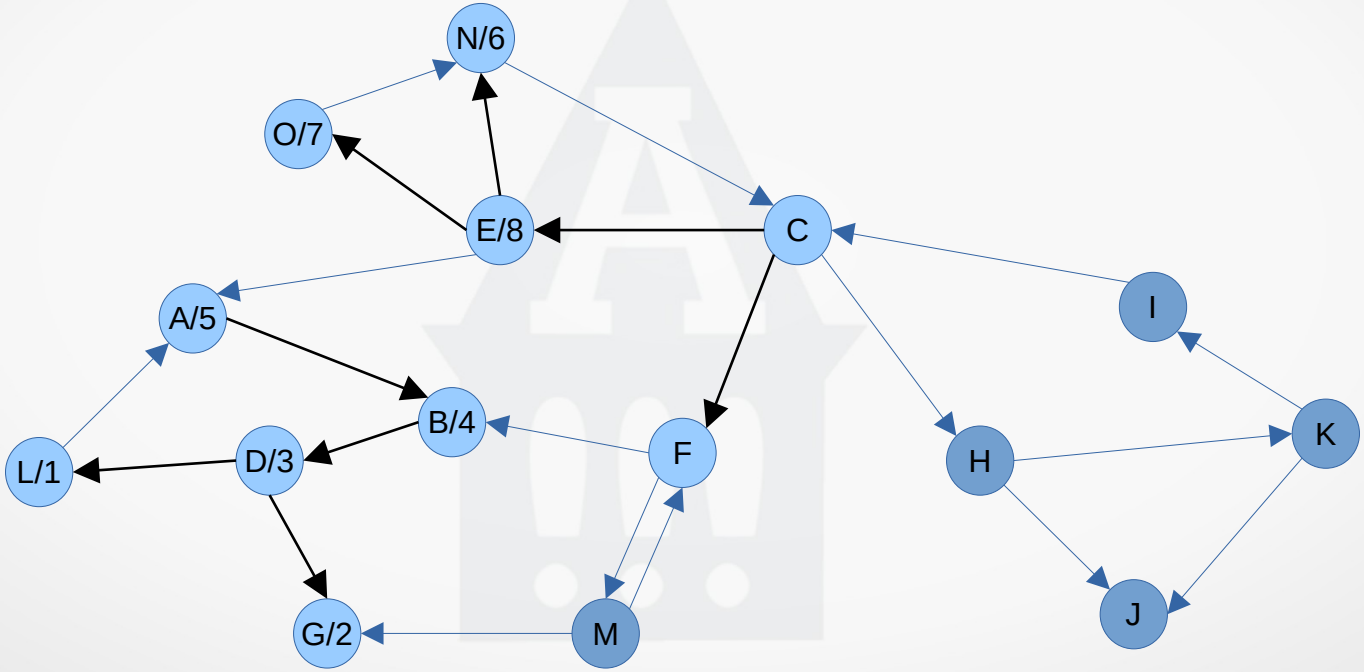
Finding Strongly Connected Components

- Start over again with C, then E, then N, then O, number E on the way back



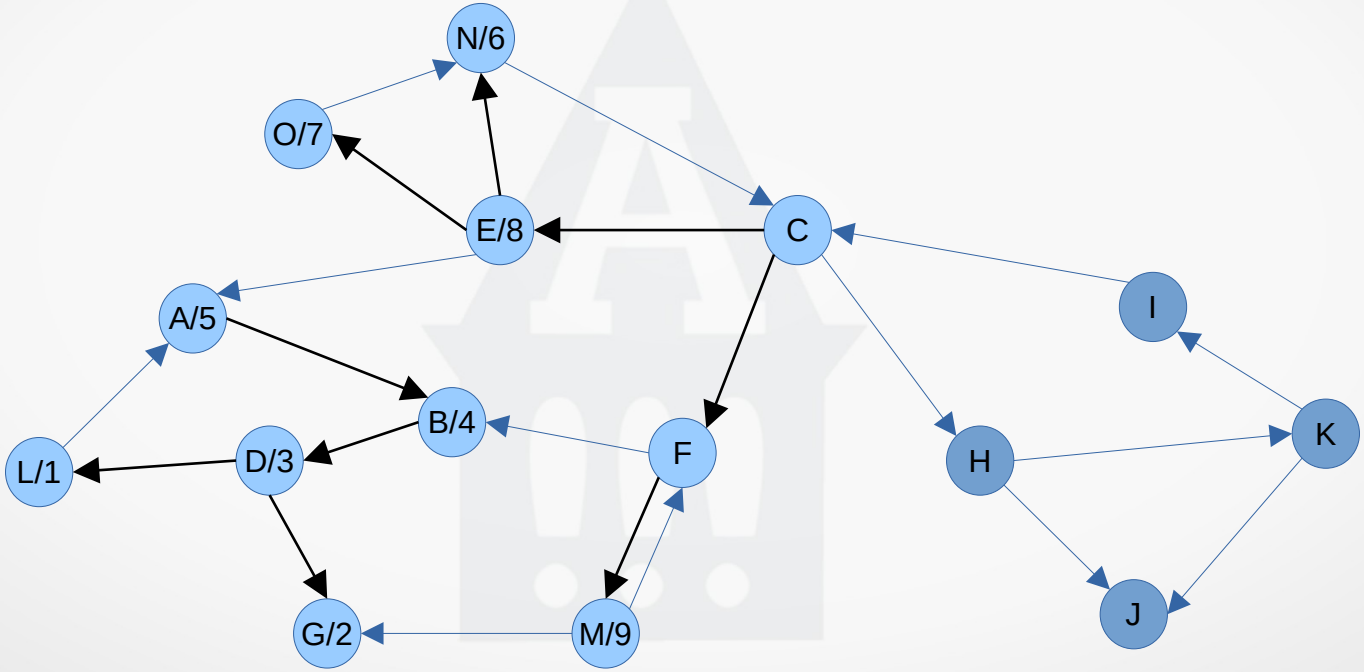
Finding Strongly Connected Components

- Back at C, then F



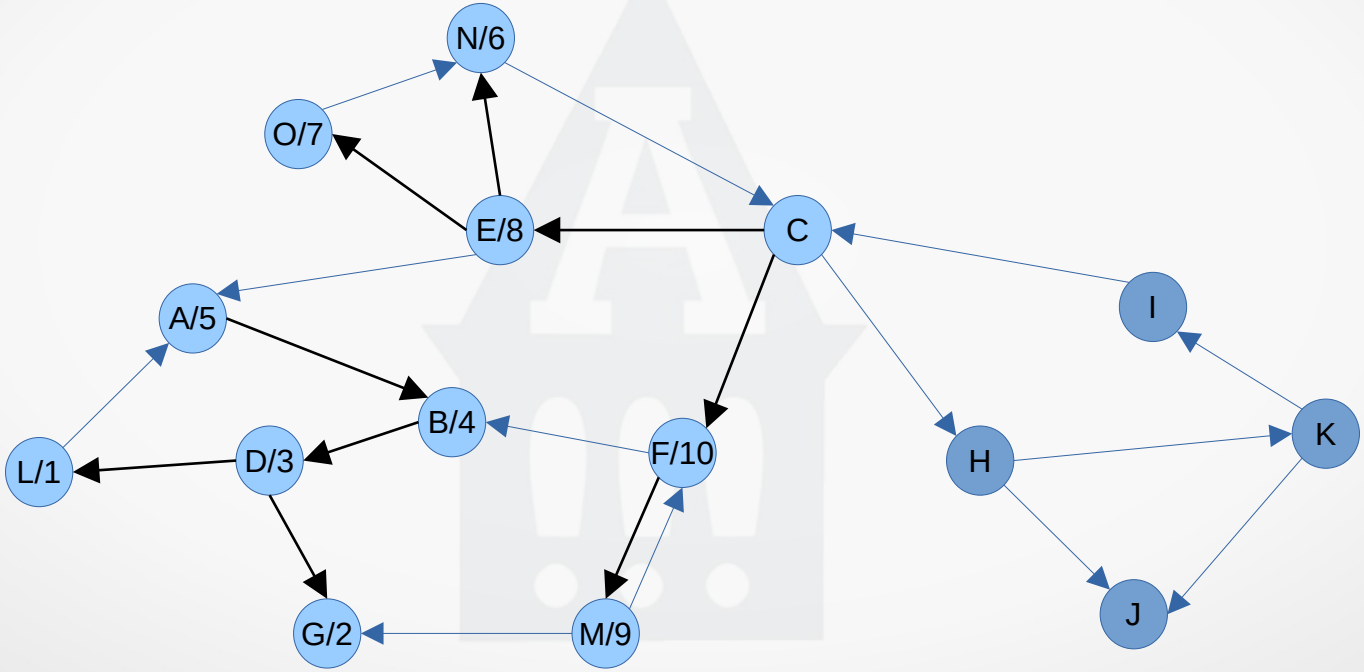
Finding Strongly Connected Components

- Back at C, then F, then M; number it



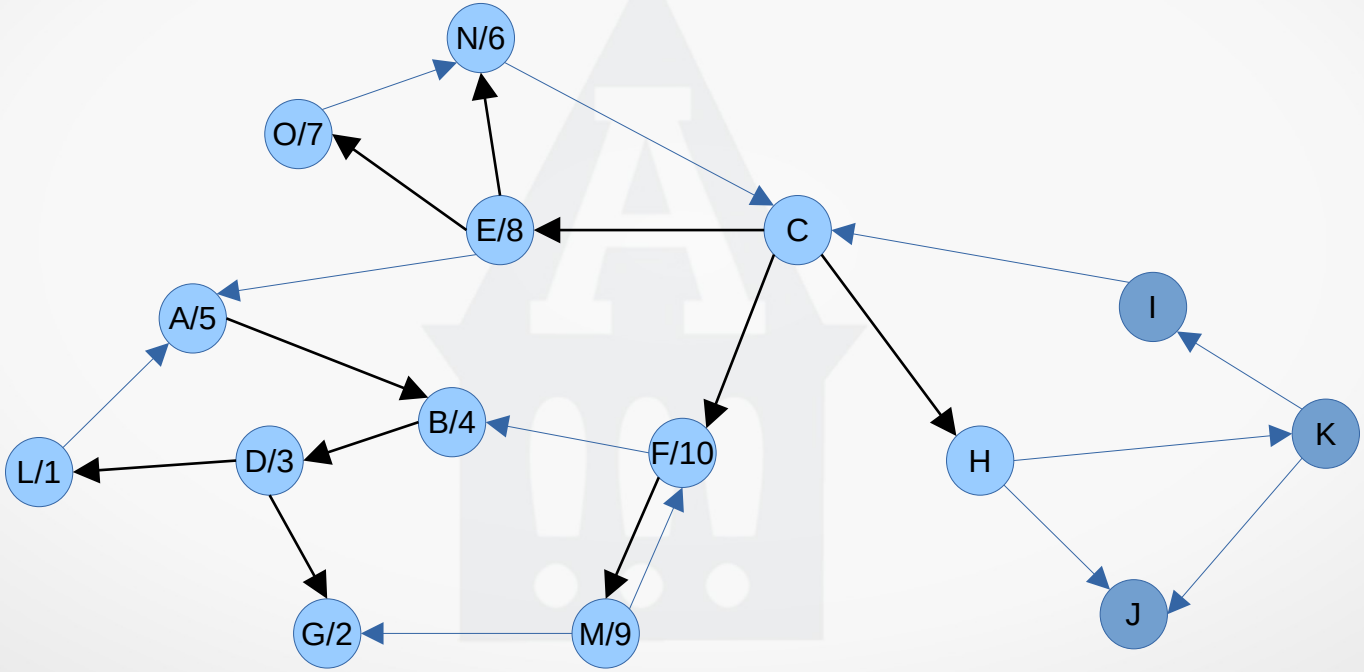
Finding Strongly Connected Components

- Back at C, then F, then M; number F on way back



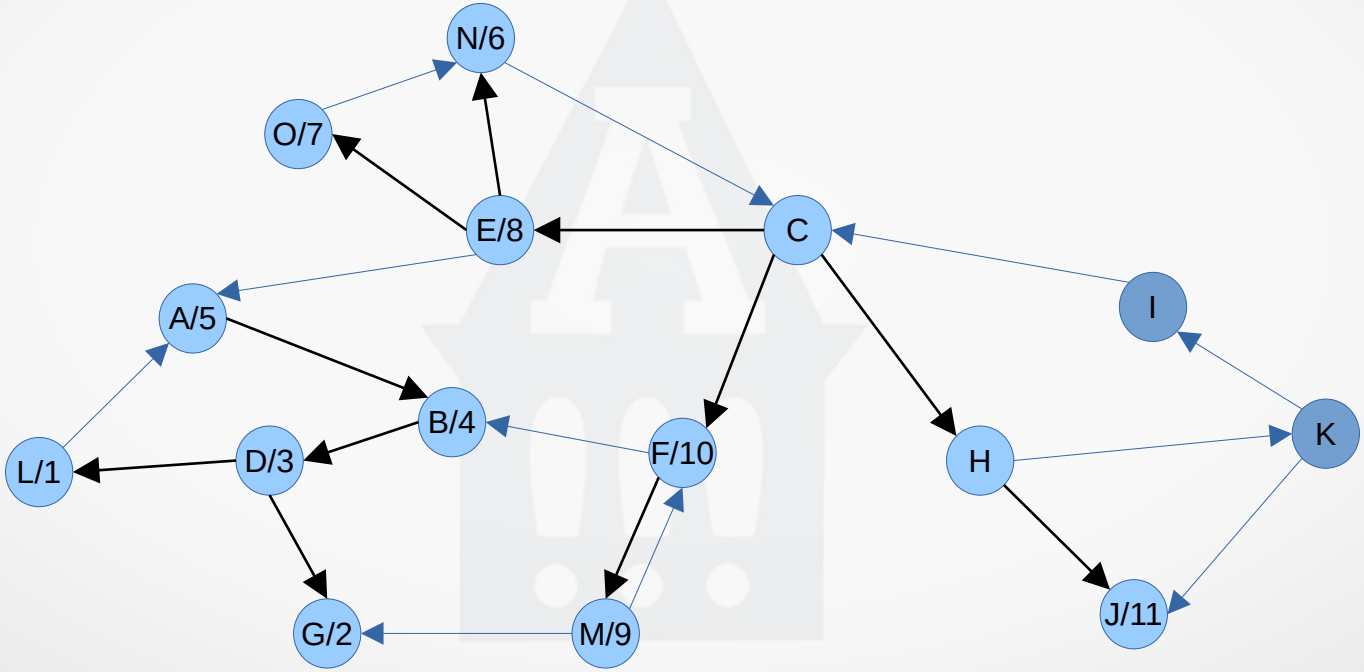
Finding Strongly Connected Components

- Back at C, then H



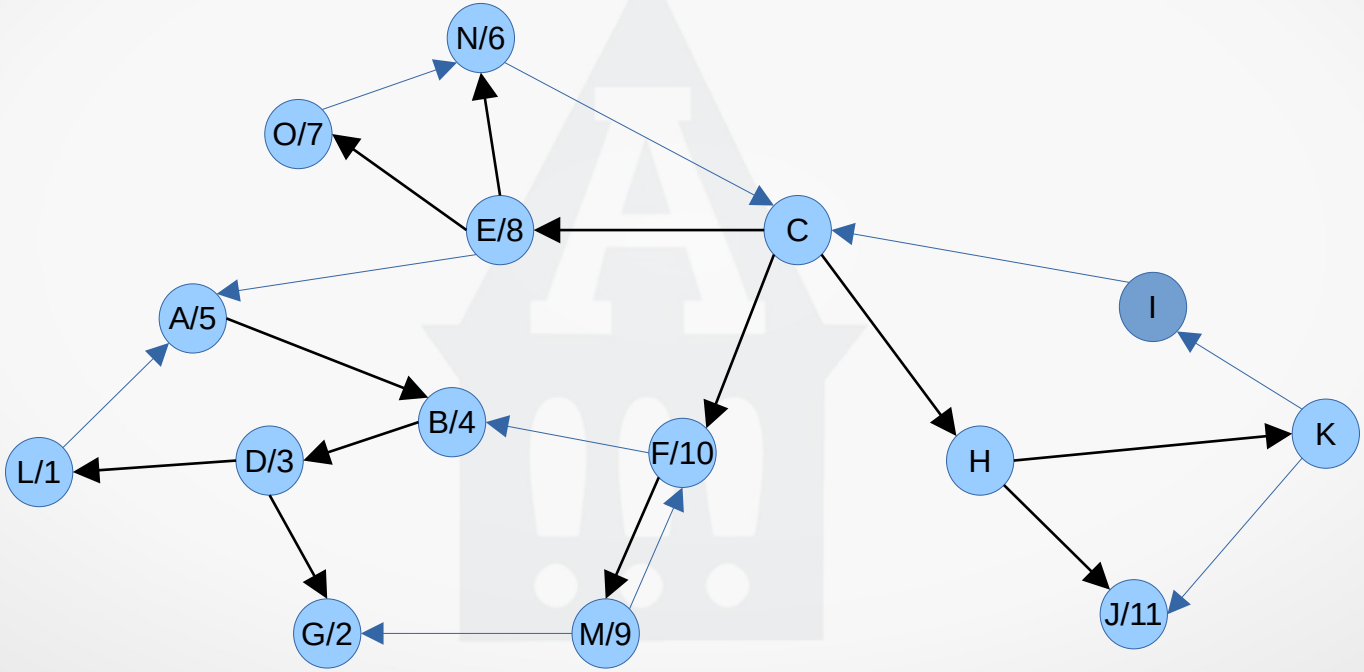
Finding Strongly Connected Components

- Back at C, then H, then J; number it



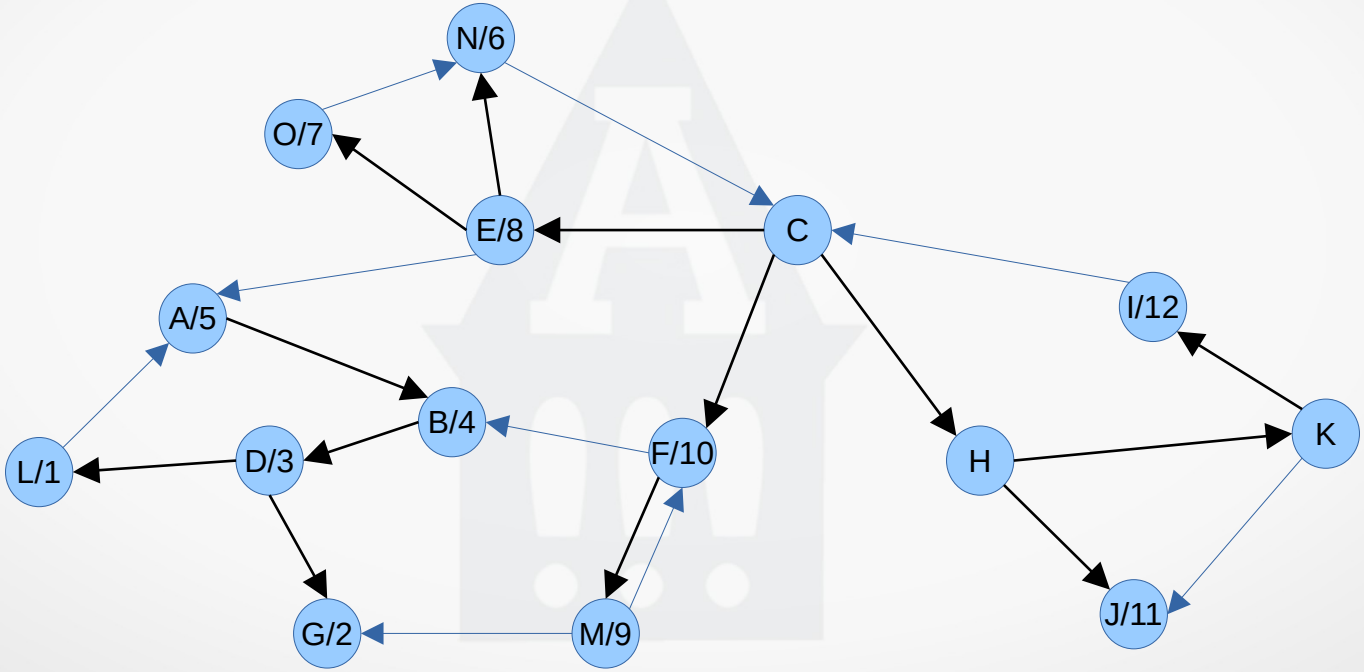
Finding Strongly Connected Components

- Back at C, then H, then K



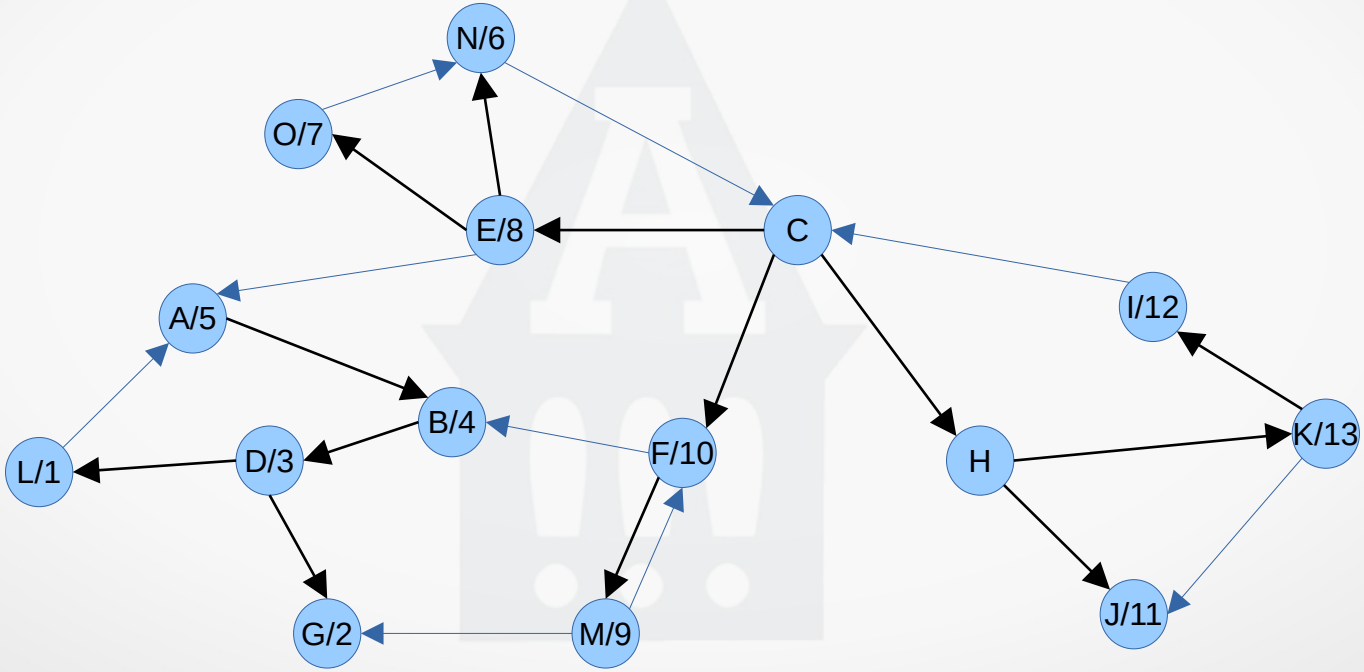
Finding Strongly Connected Components

- Back at C, then H, then K, then I; number it



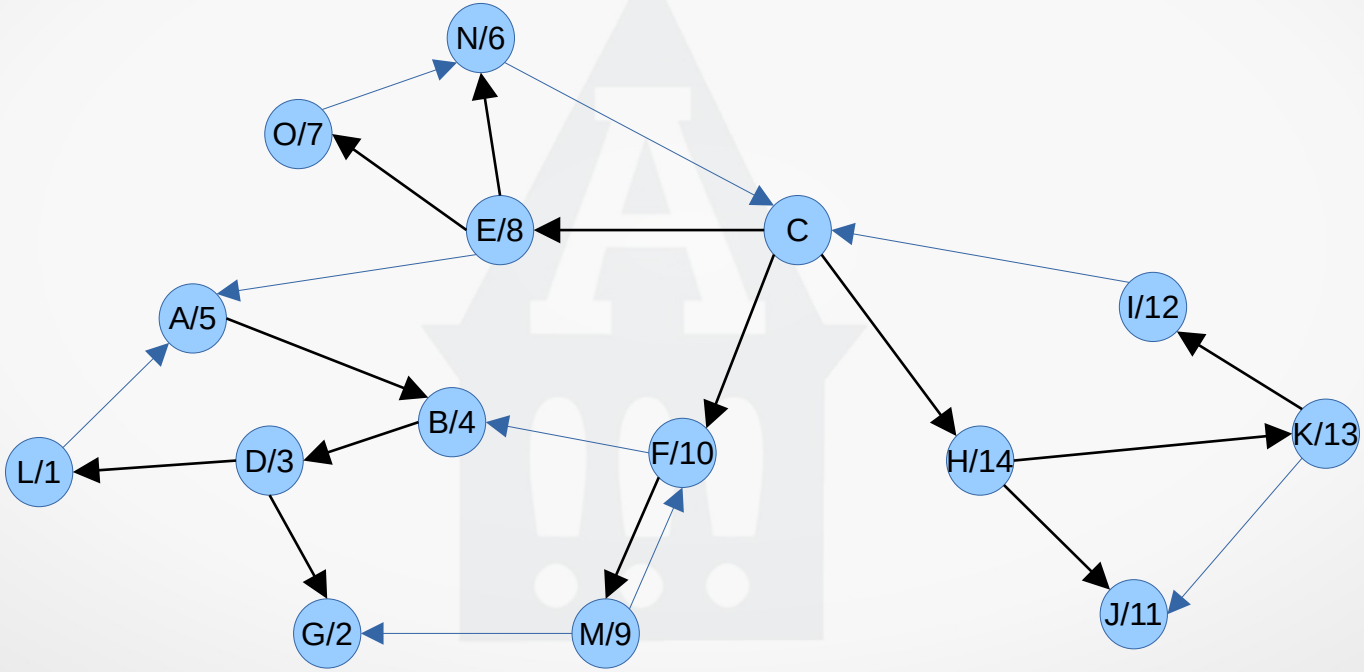
Finding Strongly Connected Components

- Number K on way back out



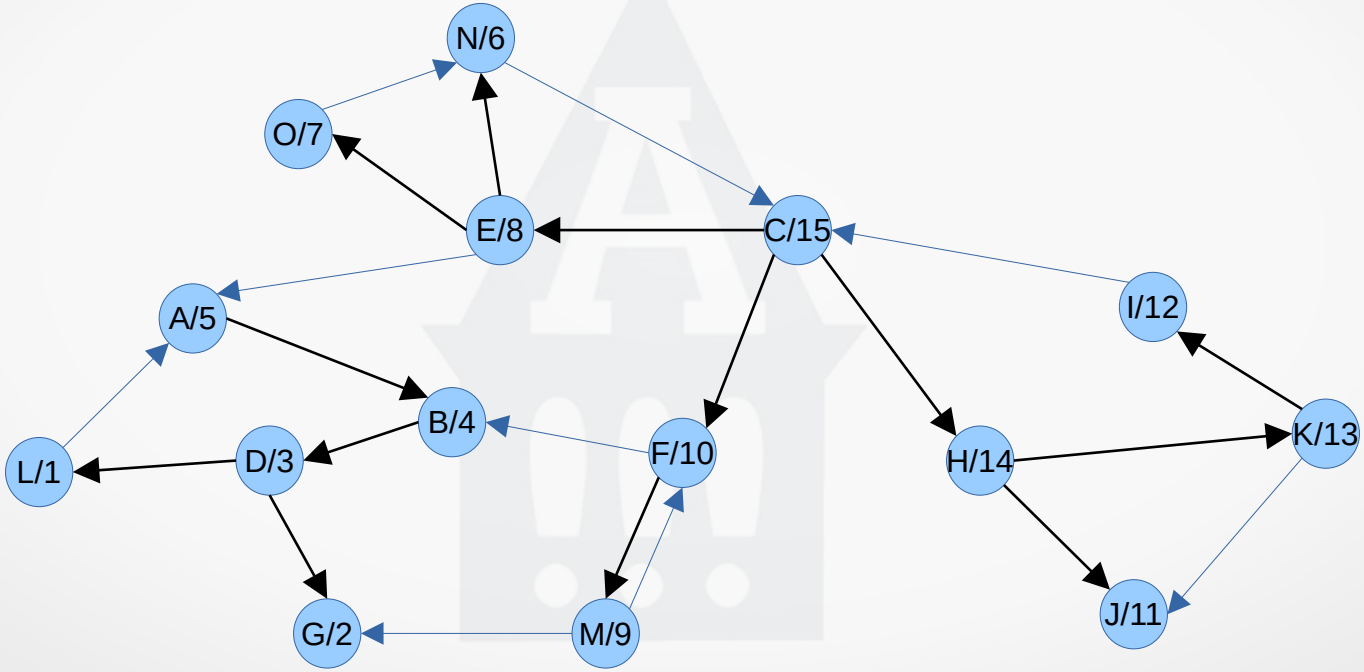
Finding Strongly Connected Components

- Number H on way back out



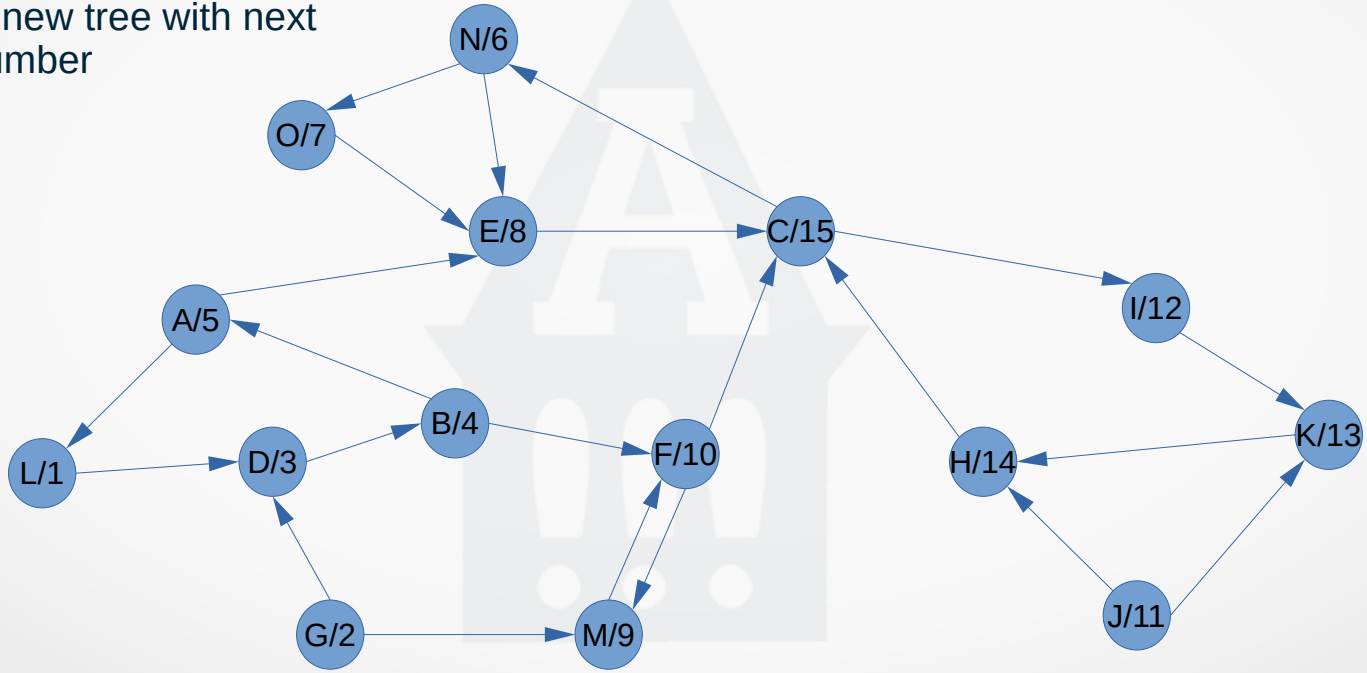
Finding Strongly Connected Components

- Number C on way back out



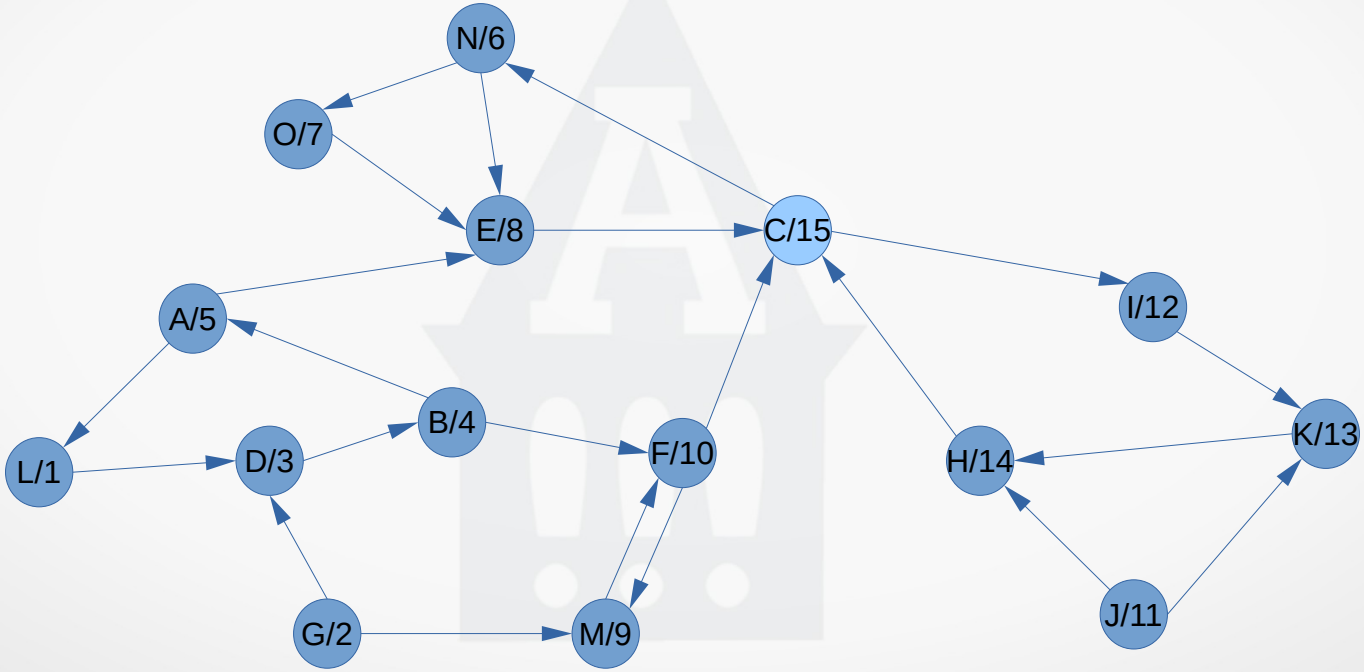
Finding Strongly Connected Components

- Reverse **all** edges, clear all visited flags, do a depth first traversal, marking nodes as visited
- Starting a new tree with next highest number



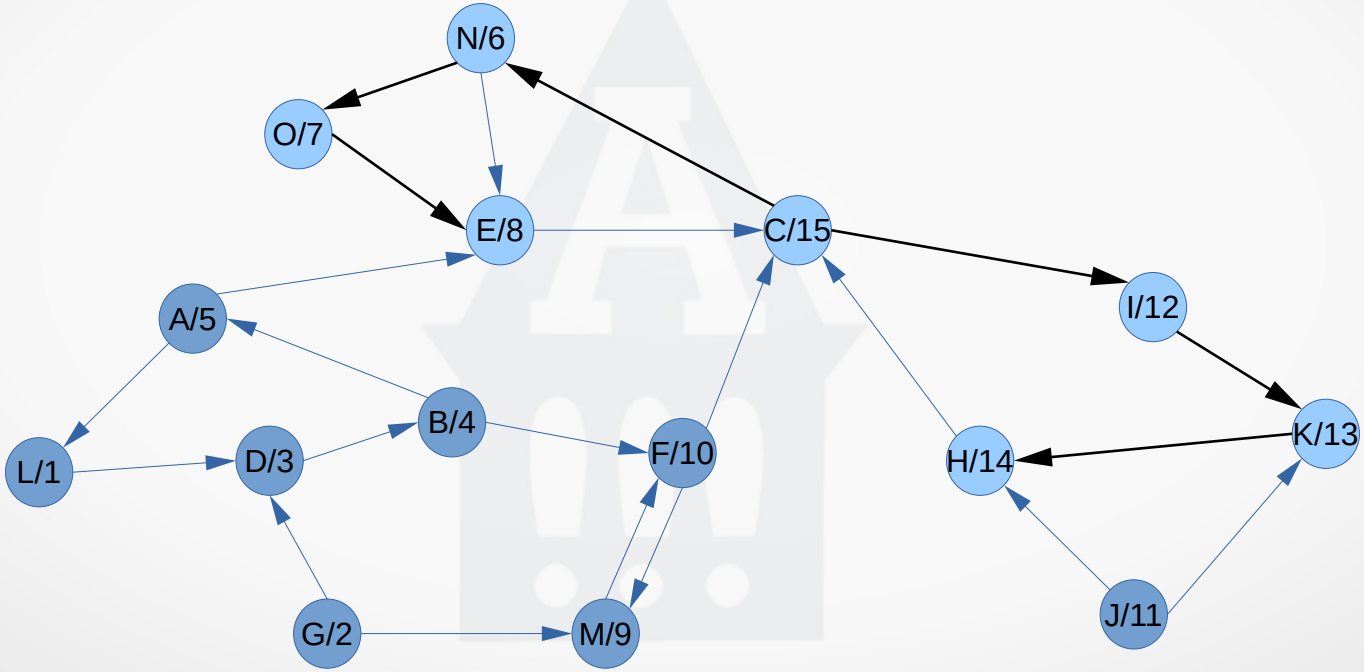
Finding Strongly Connected Components

- Start at C



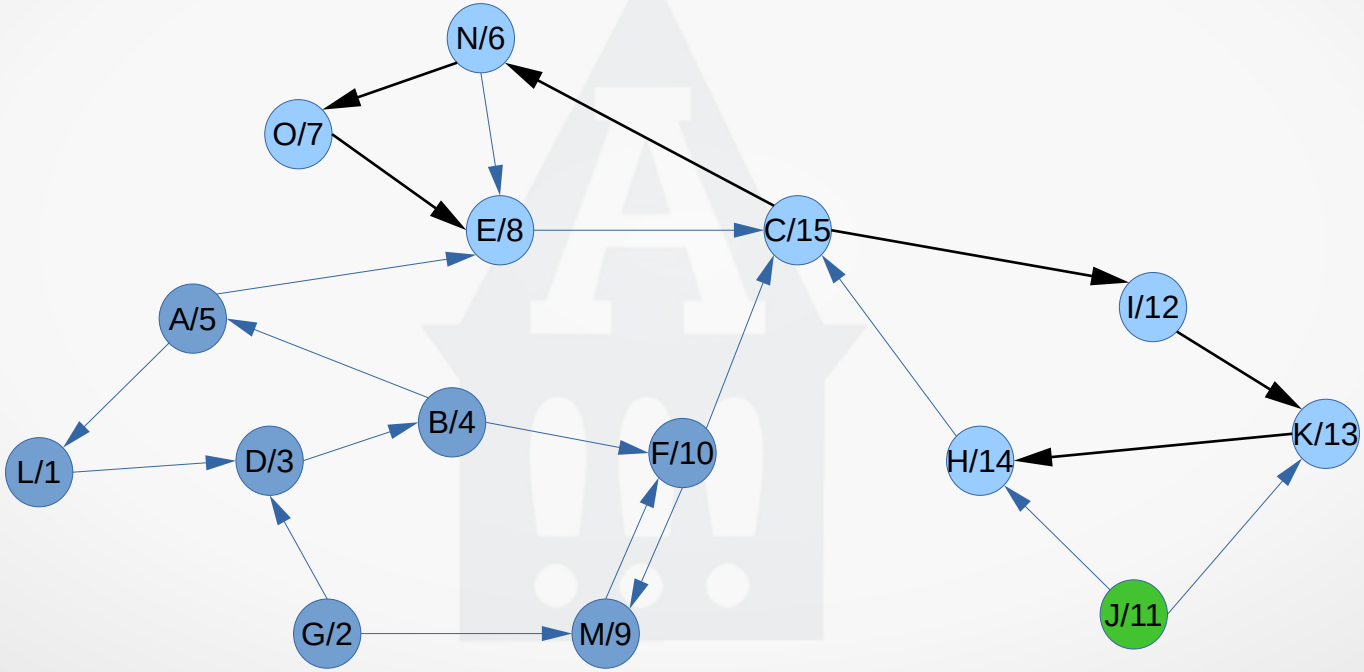
Finding Strongly Connected Components

- Start at C, and mark all nodes that can be reached



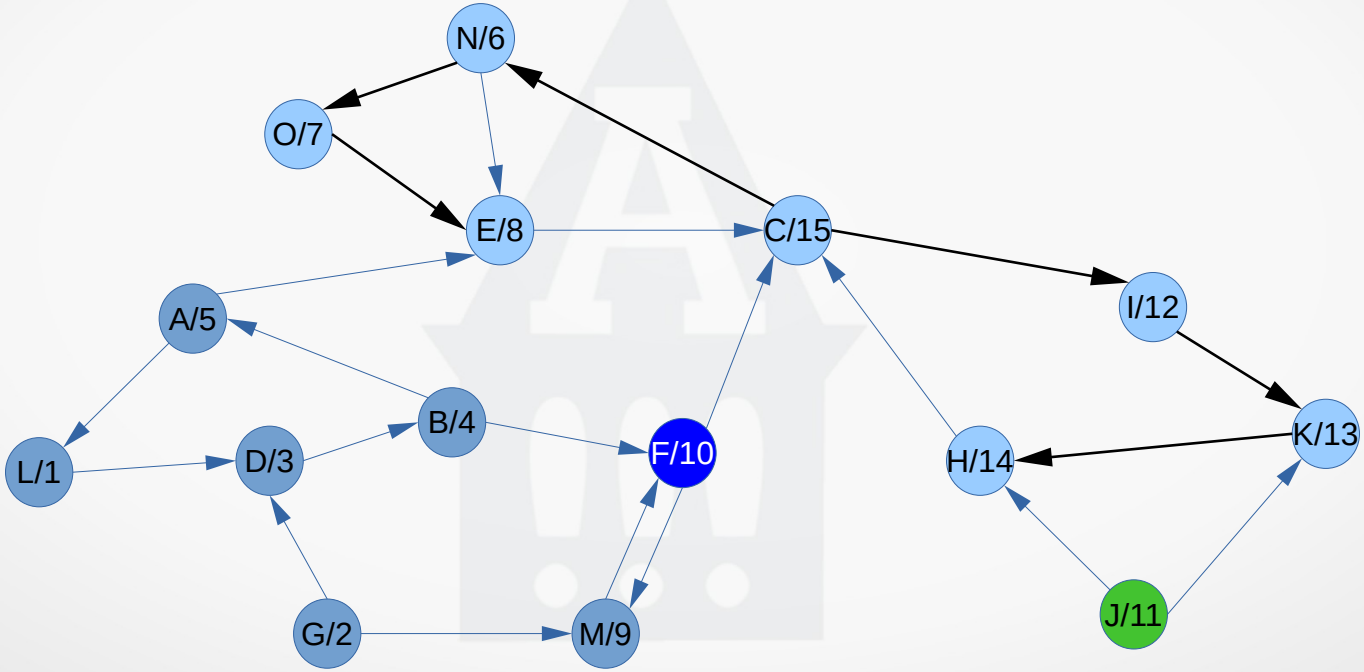
Finding Strongly Connected Components

- Start at J (next highest number); can't reach anywhere else



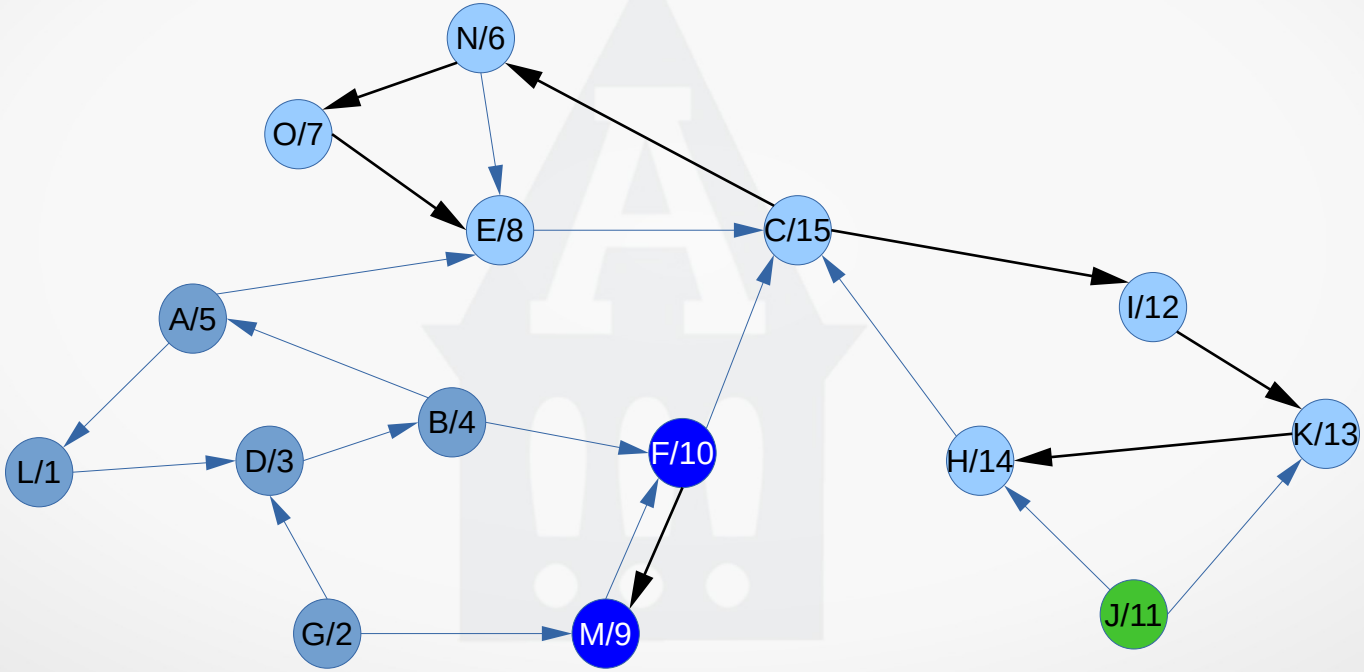
Finding Strongly Connected Components

- Start at F (next highest number)



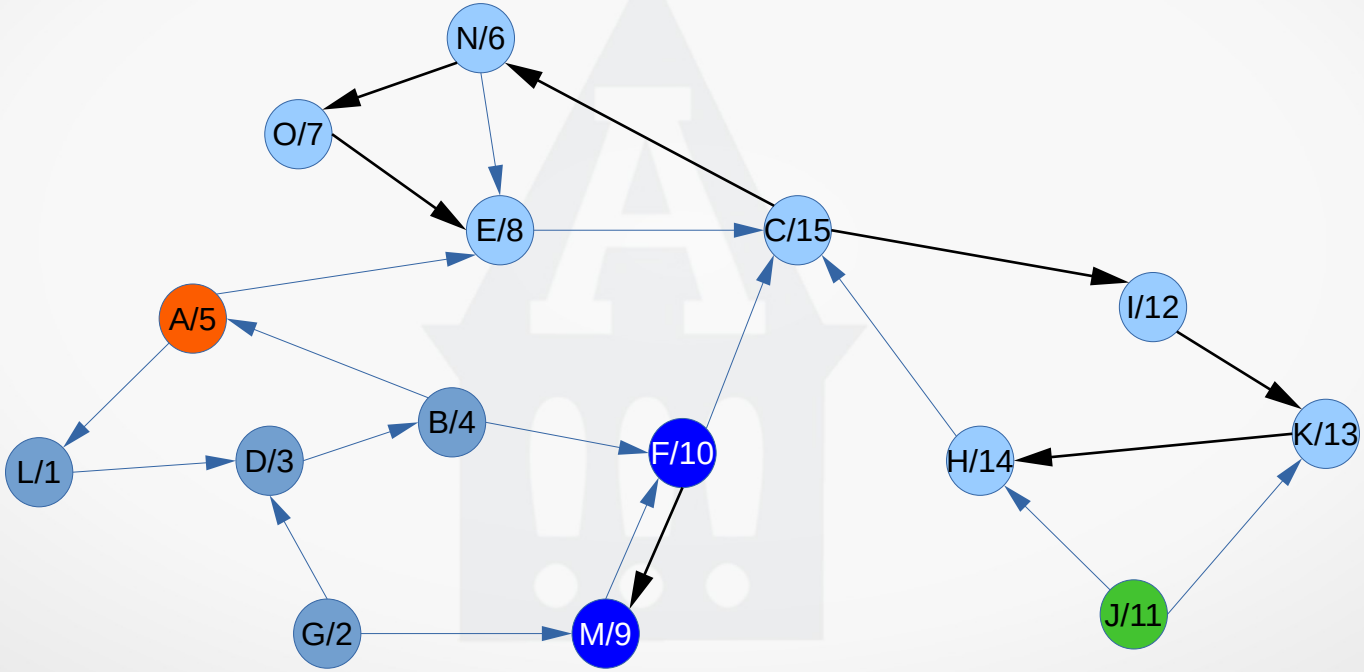
Finding Strongly Connected Components

- Start at F (next highest number); all done



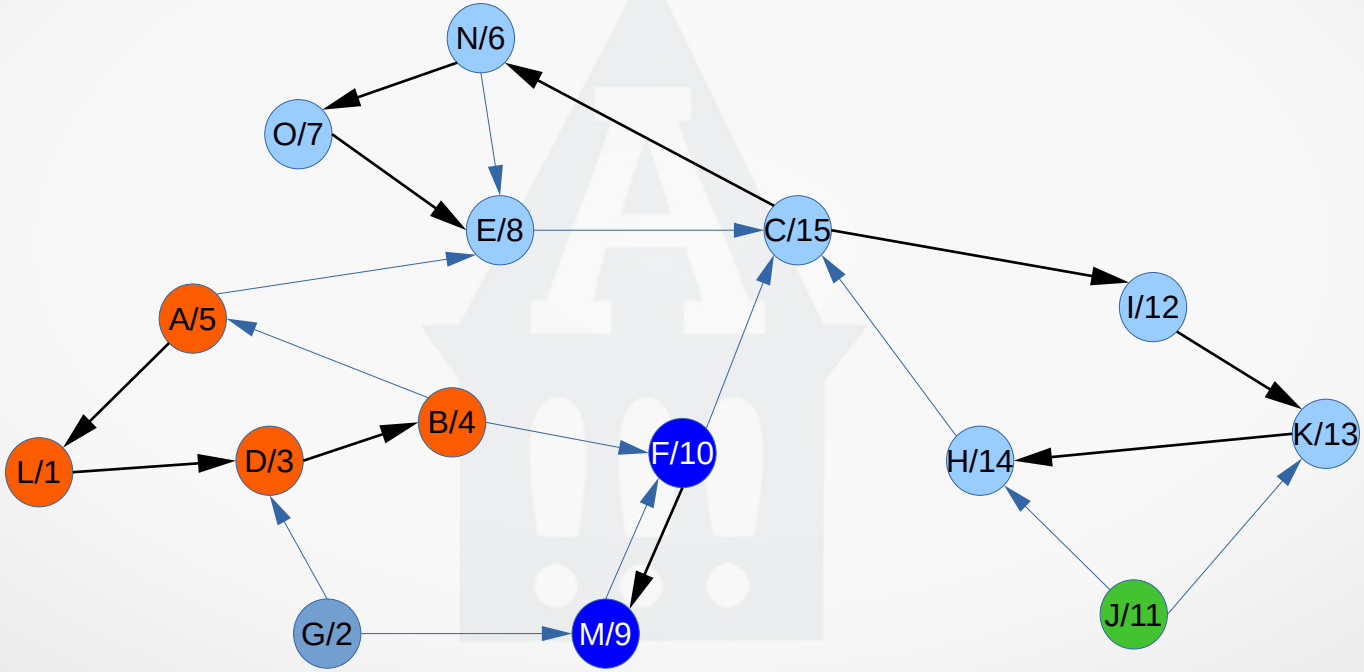
Finding Strongly Connected Components

- Start at A (next highest number)



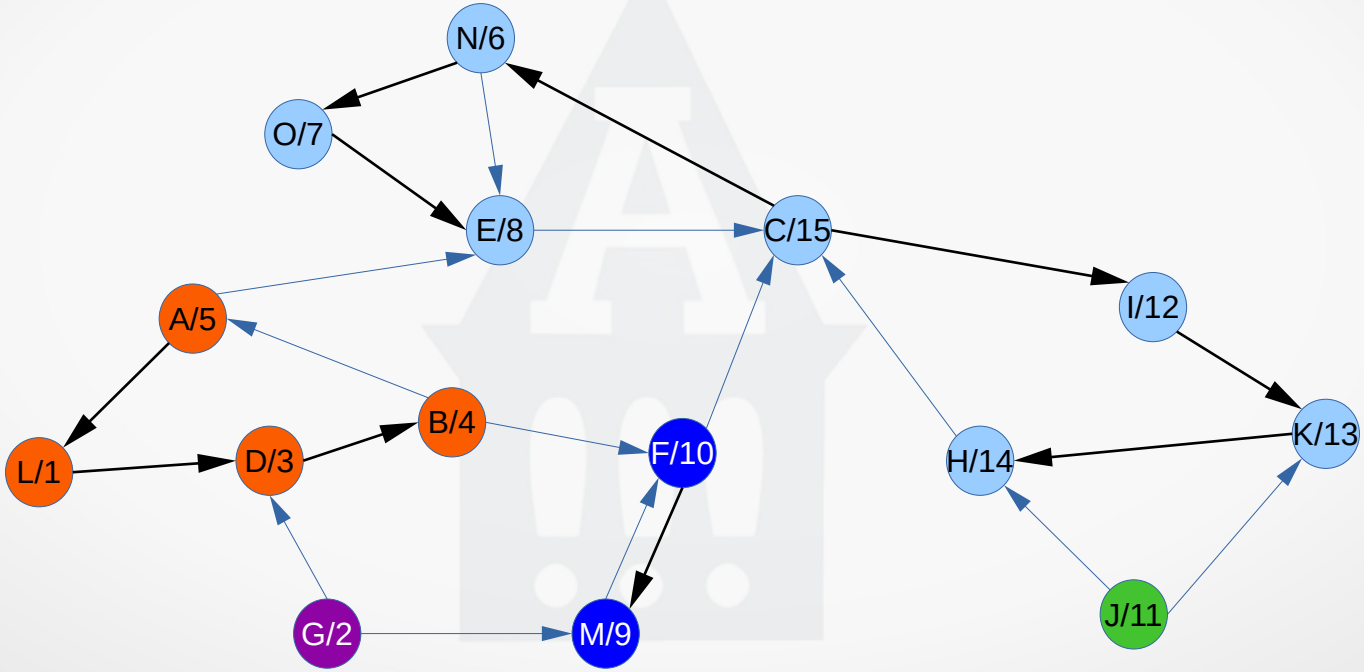
Finding Strongly Connected Components

- Start at A (next highest number); all done



Finding Strongly Connected Components

- Start at G (next highest number); all done



Finding Strongly Connected Components

- These are the strongly connected components (SCC)
  - { N, O, E, C, I, K, H }
  - { J }
  - { F, M }
  - { A, L, D, B }
  - { G }