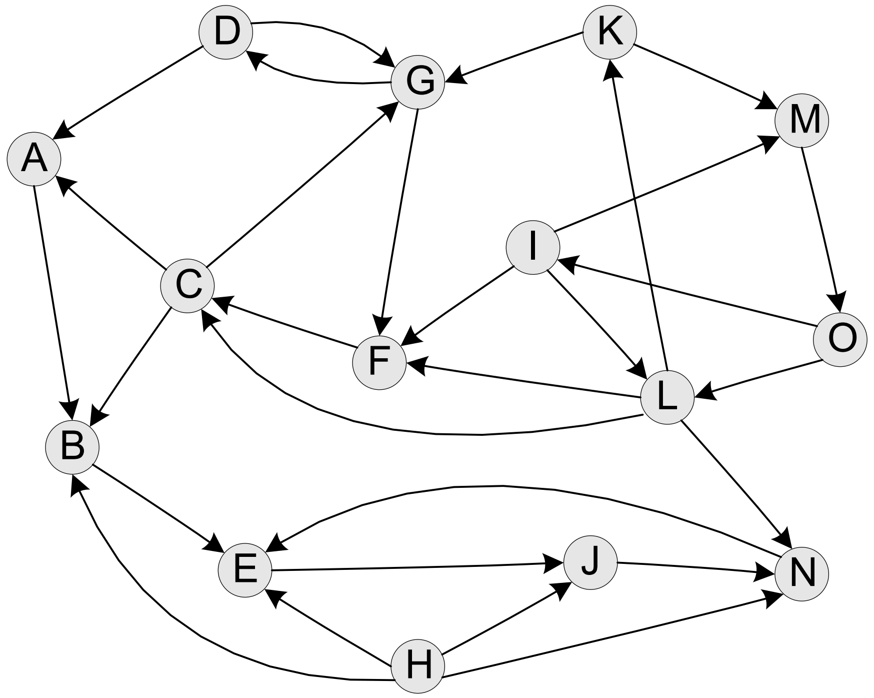
CS / MCS 401 Exercise Set #6 — Summer 2007

1. Perform a depth-first search of the digraph below, labeling each vertex with its discovery and finish times. Whenever you must make an arbitrary choice among two or more vertices, choose the vertex that comes first alphabetically. Let the discovery time of the first vertex be 1 (not 0).

Highlight the tree edges in the digraph.



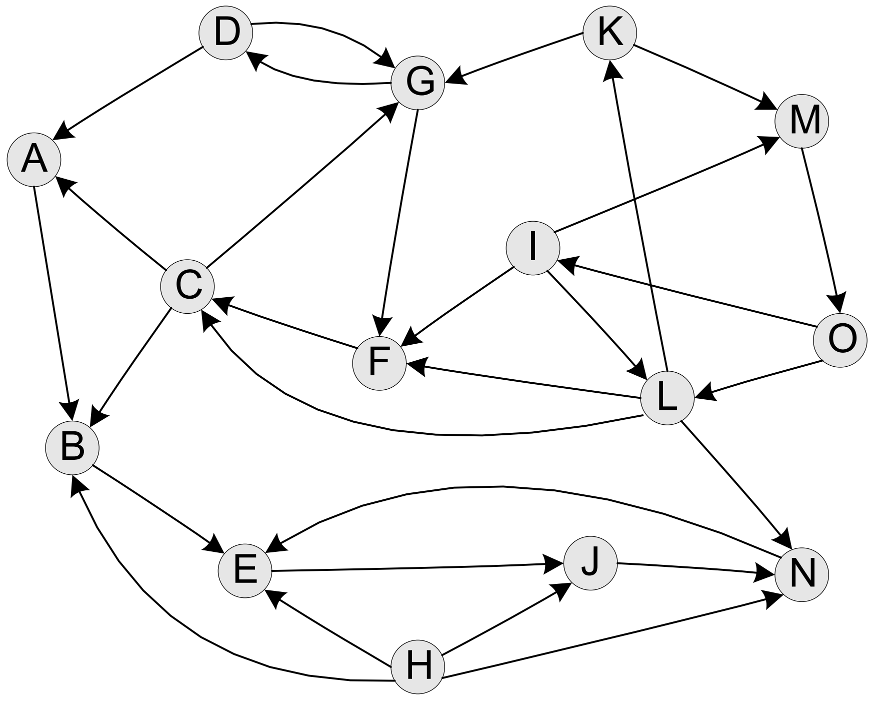
2. **a)** In the depth-first search of problem 3, which edges are back edges?

b) Which edges are forward edges?

c) Which edges are cross edges[1]?

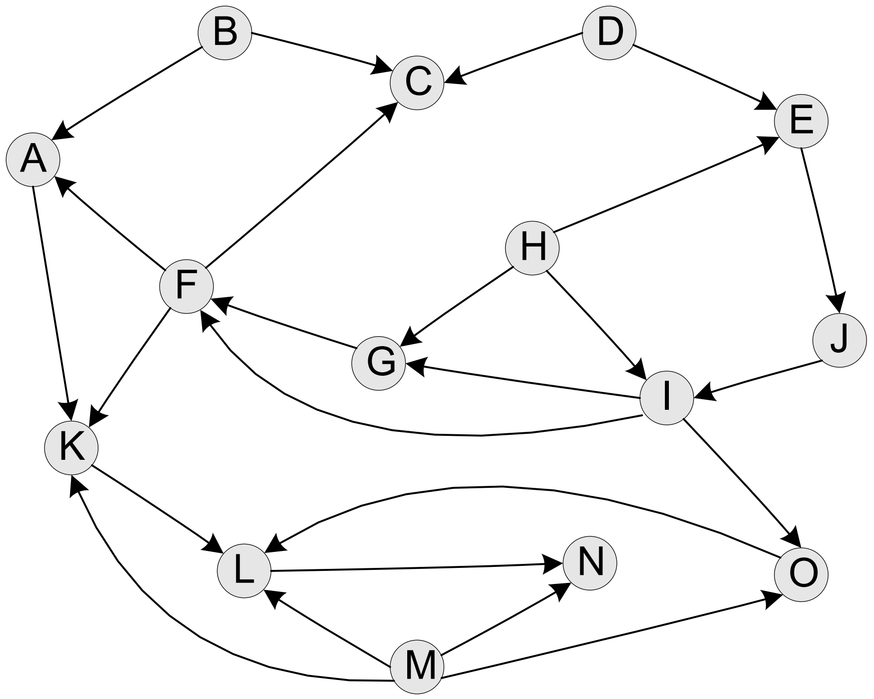
[1] The set of tree edges in a depth-first search forms a forest (a set of disjoint trees), which we shall call the depth-first search forest. If every vertex is reachable from the starting vertex, this forest is a tree. As mentioned in class, a cross edge can lead between different branches of a tree in the depth-first search forest. It can also lead from one tree in the forest to another tree, traversed earlier in the depth-first search.

3. Repeat problem 3, using an alphabetically-last rule, rather than an alphabetically-first rule for making an arbitrary choice among vertices.



4. Use the topological sort algorithm to topologically sort the acyclic digraph below. As in problem 3, use an alphabetically-first rule whenever you make an arbitrary choice among two or more vertices.

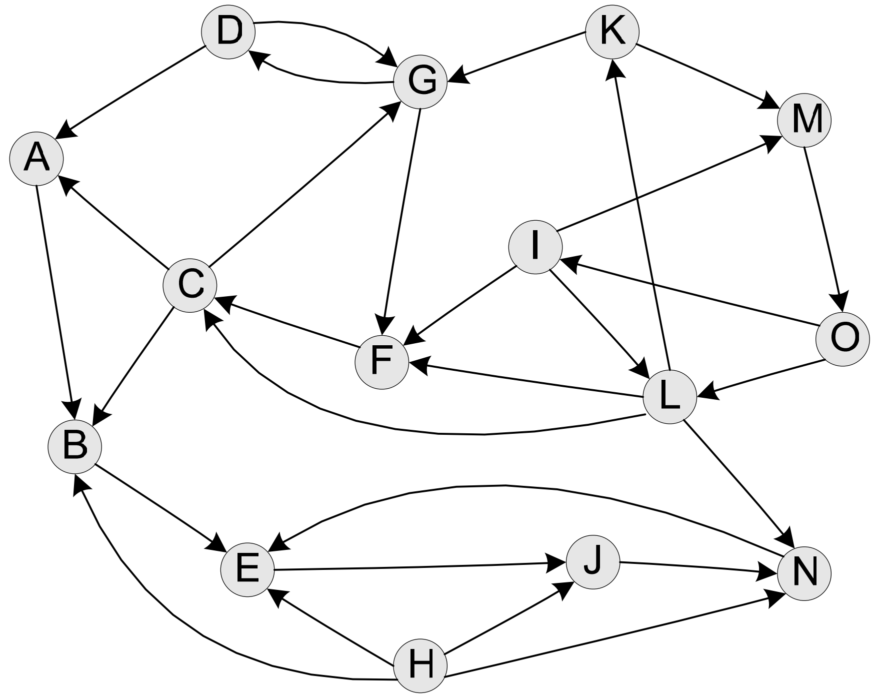
Show the label each vertex receives in the topological sort. (You do **not** need to show the start and finish times of vertices.)



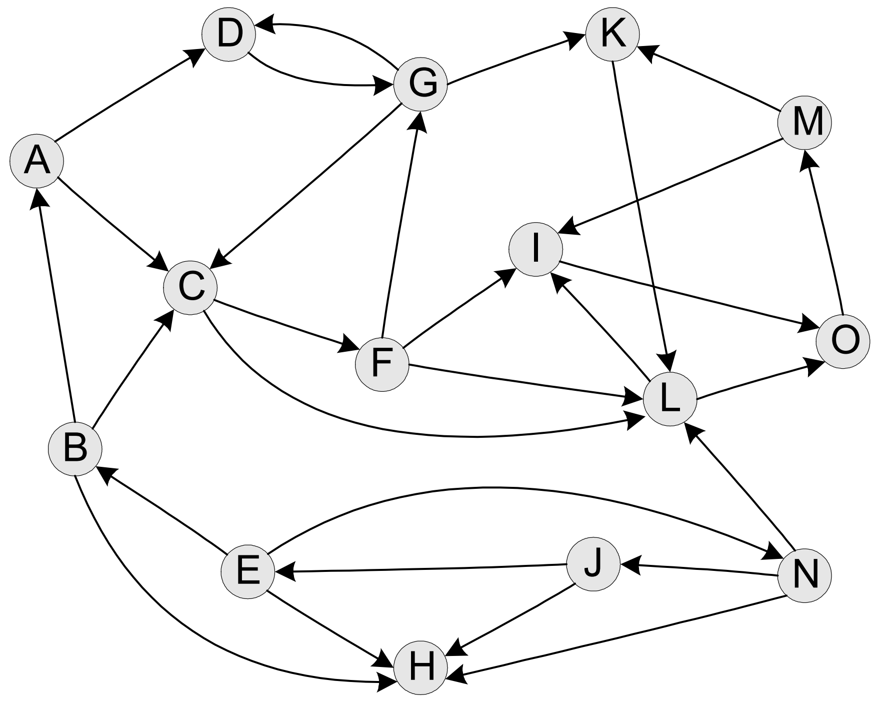
5. Find the strong components of the first digraph below. The second digraph (for use in phase 2) is the reverse of the first. In each depth-first search, use an alphabetically-first rule in making an arbitrary choice among vertices. (Note, however, that in phase 2 the choices of starting vertices are not arbitrary.)

In phase 1, label each vertex of first digraph by the label it receives (as in topological sort, except a back edge is not an error). In phase 2, label each vertex of the second digraph by the starting vertex from which it is reached, in the depth-first search.

phase 1



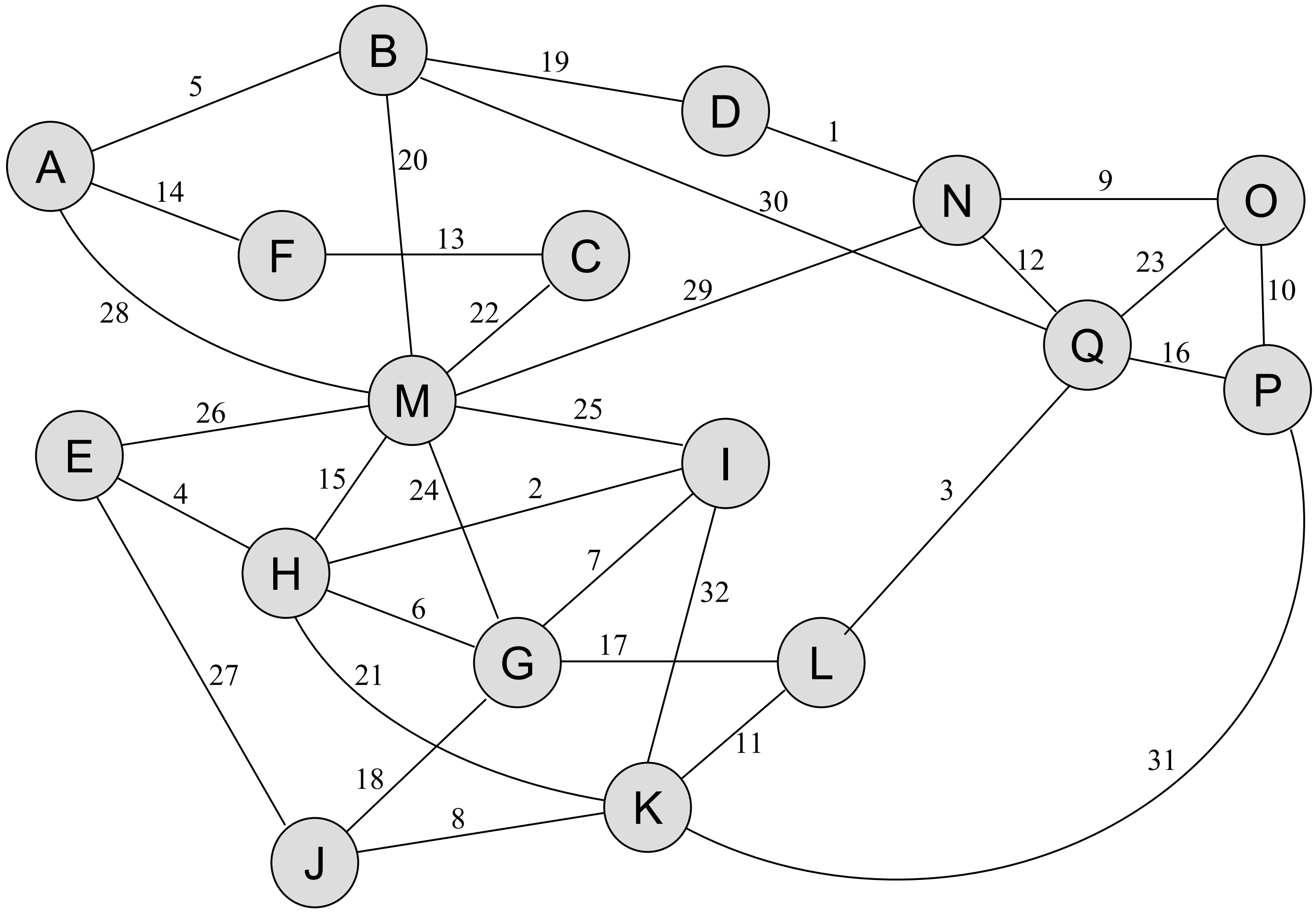
phase 2



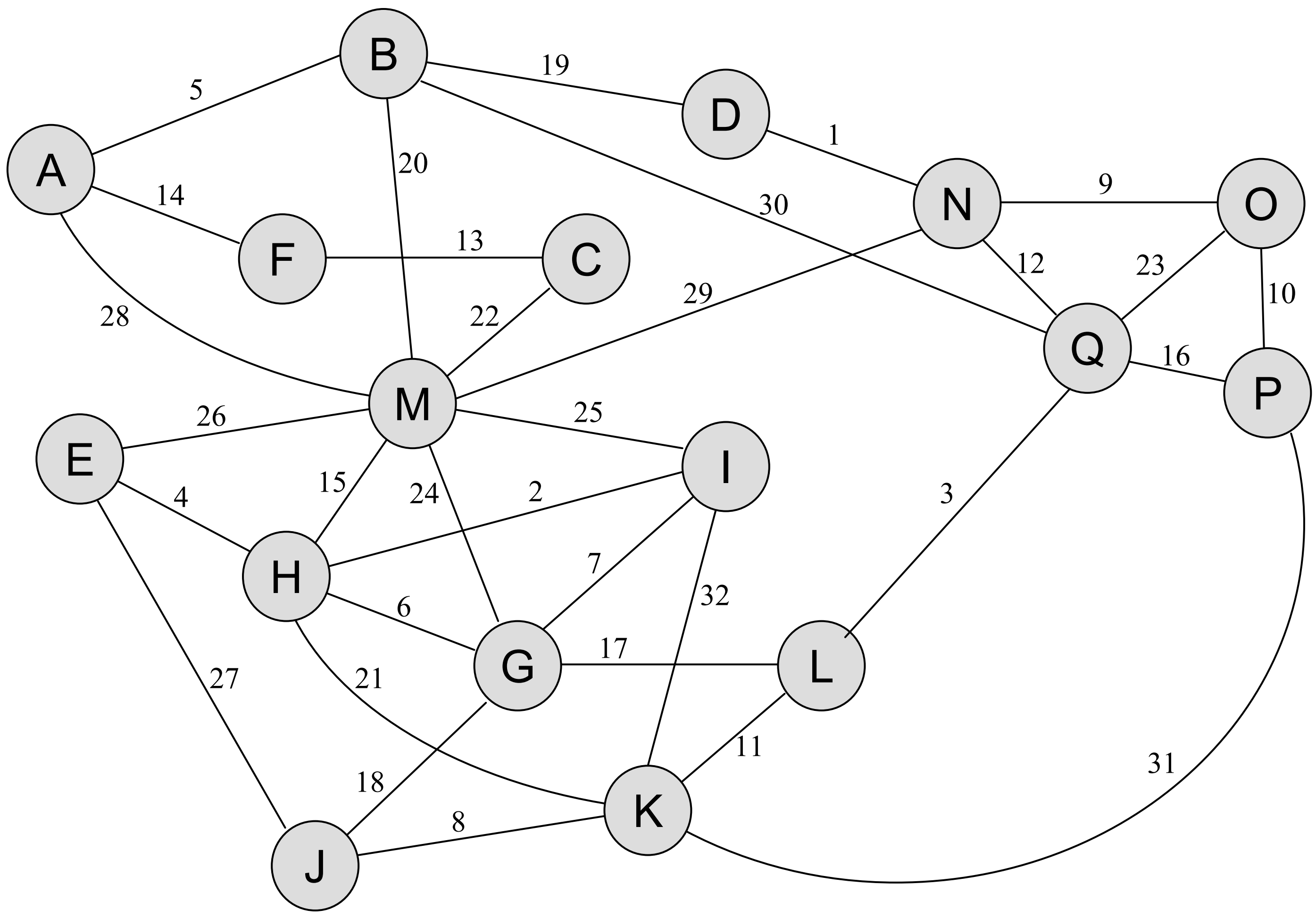
6. **a)** Suppose that, in a depth-first search of a digraph, and every edge turns out to be a cross edge. Show that the digraph must be acyclic.

b) Show that, in an acyclic digraph, a depth-first search will have only cross edges provided the order of the starting vertices is chosen correctly.

***Exercise P*:** Use Prim's Algorithm to find a minimal spanning tree for the connected, weighted graph below, starting from vertex A. List the edges of the MST in the order chosen. (Note: there should be one edge of weight *i* for each *i*, $1 \le i \le 32$; this may help a bit in carrying out the algorithm.)



***Exercise Q*:** Use Kruskal's Algorithm to find a minimal spanning tree for the connected, weighted graph below, starting from vertex A. (This is the same graph as in exercise A.) List the edges of the MST in the order chosen. (Note: there should be one edge of weight *i* for each *i*, 1 ≤ *i* ≤ 32; this may help a bit in carrying out the algorithm.)



A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
5
19
1
20
14
13
30
9
12
23
10
28
22
29
16
26
25
4
15
24
2
3
7
32
6
17
27
21
11
31
18
8