Assignment 17

Alex Clemmer
Student number: u0458675

This problem breaks down to a small set of cases that depend on the number of sinks and strongly connected components:

- If there are 2 or more sinks, we know for sure that the number of nodes that are reachable by all other nodes is 0, since no nodes in either sink can reach each other.
- If there is one sink, then the nodes reachable by all nodes is the number of strongly-connected components in that sink.

Accomplishing this is really simple (and was outlined in class). Since our graph comes as either an adjacency matrix or an adjacency list, we are going to have to traverse it. I won't supply the pseudocode for this algorithm, because I did that 2 assignments ago.

From here we do the same G^R algorithm we should now be familiar with. We reverse the polarity of all the edges and traverse the graph with a depth-first search. This runs in linear time, as we traverse each node and each edge once.

From here we again reverse the polarity of the graph and begin exploring, sorting the next nodes by the post number in ambiguous cases. The number of time we reach the outermost level of recursion is the number of sinks components that we have. This is because, when we reach the end of a sink, we end up recursing all the way back to the top of the recursive tree.

IMPORTANT: The total sink count cannot be more than 1. Also, when we have found our one sink, simply breadth-first searching it will give us the total nodes that all nodes can reach.

Reversing polarity and finding all nodes connected to one node in a graph are all $O(n)$, so if we're operating via an adjacency matrix, this will not alter the $O(|V|^2)$ running time. Likewise, a linear run time adds nothing to $O(|V| + |E|)$.