Assignment 21

Alex Clemmer
Student number: u0458675

Finding the shortest path between a one-source, one-sink graph is conceptually pretty simple. First, find the strongly-connected components using the G^R. That is, you start by depth-first searching, applying pre- and post-numbers as you go. Immediately as you start recursing back, you've found your sink. Find all strongly connected components here through a breadth-first search. This is linear.

Each point we encounter a node, we add the node to some (hash?) table. Every node we find in the strongly-connected components is pulled off the table as we find them. We can repeat this via the post-numbers to find the connected components of the source. These are both linear.

From here we can perform the shortest-path single-source DAGs algorithm on page 119. Basically, since the DAG is linearized, we can avoid the overhead of the priority queue in Dijkstra's. We restrict our domain to all the nodes that are not in either strongly connected component. This should also be linear.

Since everything is linear, the end result should also be linear.