Gray Houston

ghousto

0026483532

Note: Throughout this analysis, the variable *n* will always refer to the number of nodes/vertices in the graph and the variable *m* will always refer to the number of edges in the graph.

Project 5 Analysis

1. The asymptotic performance of my implementation of the topological sort algorithm is O(*n* + *m*), where *n* is the number of nodes in the graph and *m* is the number of edges. In my implementation, the source nodes (i.e. nodes with in-degree of 0) are identified when the graph is read in from the file. Therefore, determining the source nodes will not contribute to the algorithm’s performance/running time.

Assuming that the source vertices have previously been identified and stored in a list, my implementation performs a BFS-esque traversal of the graph. When sources nodes are removed from the graph and added to the topological ordering, only that source node’s children (i.e. the vertices pointed toward by the source) are checked if they have become “relative” source vertices.

By only checking a source node’s children for new sources and only processing these new sources, each node is accessed and processed exactly once. This traversal of every node exactly once while creating the topological ordering produces an O(*n*) asymptotic performance.

In addition, the number of nodes traversed by looking at all of a source vertex’s children for all relative source nodes is equal to *m*. Proving that every edge is traversed exactly once is slightly non-trivial:

All source nodes only contain out-bound edges

When a source is removed from the graph, each out-edge to a child is removed

Right before a source is deleted, each out-edge is followed to its children

Therefore, all the out-edges of a deleted source node are traversed exactly once

All of nodes of the graph are eventually removed as “relative” source nodes

Therefore, the out-edges of all of the nodes are traversed exactly once

The total of all out-edges in the graph is *m*

Therefore, the total number of node access that occur by checking a source’s children for new sources vertices runs asymptotically in O(*m*) time.

Furthermore, because the O(*n*) traversal to create the topological ordering is not directly related to the O(*m*) traversal used to check a source vertex’s children, the performance of these two processes is summed instead of multiplied. Therefore, the overall asymptotic performance of the algorithm is O(*n* + *m*).