John Carrabino
December 4th 2016
CS 261 Assignment 67

Graph Traversals

1. How is the graph stored in the provided code? Is it represented as an adjacency matrix or list?

The graphs provided are stored using an adjacency list, because they store only the edges/arcs adjacent to each vertex instead of a 2d matrix containing the adjacency of each vertex with every other vertex in the graph.

2. Which of the 3 graphs are connected? How can you tell?

Graphs 2 and 3 are both connected. You can tell this because after running prog the output for both graphs show that each vertex is connected to every other vertex after running both a DFS and BFS search.

3. Imagine that we ran each depth-first and breadth-first searches in the other direction (from destination to source). Would the output change at all? Would the output change if the graphs were directed graphs?

If we ran both searches on an undirected graph, then reversing the path of traversal would not alter the output since the resulting edges can all be crossed in either direction. However, if the graphs were directed graphs then the output would change because all of the edges would now only be accessibly from their starting point to their ending point.

4. What are some pros and cons of DFS vs BFS? When would you use one over the other?

A DFS will mirror a search like how a human would solve a maze by forming a singular path and backtracking whenever finding a previously visited vertex. More likely than not while conducting a DFS search, backtracking is unavoidable, but if you get lucky it may be able to find the path in a single go. On the other hand, BFS will search all possible paths simultaneously. This causes the BFS search to require more memory than a DFS search, and it also means that a BFS search will visit all reachable vertices between the start and destination. If there were relatively few edges than I believe a DFS would be preferable due to its more direct approach and lower memory requirements. However, if I were searching through a graph with a larger magnitude of vertices/edges than I believe a BFS would be more efficient to use as it will examine all possible paths simultaneously.

5. What is the Big O execution time to determine if a vertex is reachable from another vertex?

The Big O execution time for both DFS & BFS will be O(E), where E is the number of edges/arcs present in the graph.