Comp 363

Dr. Sekharan

Christopher Elliott

Homework 7

2) To find all the vertices that are not connected on the graph would not be too hard for the way I implemented my data structures. One of the hardest parts was deciding what I should use to do the project. I ended up deciding to use an array of HashMaps in Java. The vertex was put into the array of 1,000,000 size at the exact index that corresponded to its value. Then at that index, a hashmap containing all the keys that the value at that index connected to on the graph is present. Therefore, if I wished to find all vertices that are not connected on the graph, I would simply have to check if the position at the index is null. This would mean that at that point, there are no connected vertices to it on the graph.

3)The best case scenario for time complexity is if the two given vertices are a pair in the graph because I used containsKey which can find if two vertices are a pair in O(1) time. However, if they are not a pair, it will keep on going through every possible pair that might connect the two vertices. Therefore the time complexity would look like O(V\*N) where V is the number of vertices that have a connection to the root vertice because it is essentially Breadth-First search. This means that the total time complexity is O(E) where E is the total number of edges that connect to and have a path to the vertice X.