Report 1
Lab 1; Exploration of path optimization
104349 Grafos, Topología y Geometría Discreta
Thijs Rood, 1644492
Joel Tapia Salvador, 1638962

17/02/2022

In this lab we tested two algorithms to search to the network of a graph for separate components. The algorithms used are breadth first search (BFS) & depth first search (DFS) respectively. Later we also explored the degrees between nodes and the diameter. The dataset is a graph network from social media platform LastFM. The questions will be answered thouroughly in this report.

Report Questions I

1) We made first ten test based on eleven based on random seeds that make different graphs. The graphs are a lot shorter than the graph from the csv file from LastFM so it would run a lot faster. Six of them have 11 nodes, one 35 on 100 and one 5. The test compares the results from the networkX library with our library and see if they are the same. It for example compares the length of the amount of components. It also draws the graphs for us so we can compare. With these tests we found for example that graphs with cycles the nodes were sometimes appended multiple times. The graph with 100 nodes it made multiple components so we were sure the amount of components function workes. (For more information visit “test_lab1_documentation.pdf”)
2) In the LastFm dataset there is only 1 component. Meaning everyone node in the graph is somehow connected.
3) When the target items are closer to the source, in a more tree-like or breadthened structure, BFS is logically faster. When targets are further from a source in a more linear structure, DFS is faster. This is because both algorithms check the nodes in a horizontal and vertical path respectively.
4) For the particular case of the LastFM dataset the DFS is (1.4x) faster. This might be explained by the large amount of connections some of the social network’s users have. This can be determined by putting a time object in the python code, it can measure the time needed to perform one of the algorithms. As the time can differ each time we run the algorithm 10 times and took the average time from both algorithms.

Report Questions II

1) We used Dijkstra's algorithm in the how_many_degrees function as it would befaster because it can be stoped once we reache node B and doesn’t requeire calculating all the shortest path every time, thing that is slow.
2) Lastfm has only one component and it’s diameter is 15.
3) We use the botton triangle of a matrix (because the graph is not directed and its matrix is symmetric), with the distances from each node, that starts with 1 if it is connected and infinity if not connected. Then we will iterate over the matrix k times where k is the number of nodes and in every iteration we go from the lines and check if there is a shorter path. This is basically floyds algorithm. After the matrix is finished we have all the shortest paths. Then we will look for the maximum number in the matrix. If the graph has more than one component the maximum element of the matrix will be infinity, so we do not apply the algorithm.