Report

This report demonstrates my flow of network in which I created from my selection of algorithms and data structure. The total of nodes which are connected to their weight are calculated to the max flow while also calculating the time complexity of the executed algorithms.

The algorithms utilised was the ford Fulkerson algorithm which was implemented to resolve the max flow issue in the network using paths which involves edges, nodes, and weights to calculate the duration of the max flow of network running. The results of the duration times are accomplished through calculating the closest shortest path to an ascending order. The breadth first search also known as BFS was utilised to find the shortest path using a queue.

As Linked list is a dynamic data structure, I utilised it therefore each node holds a reference and data for the next node on the list also allowing the memory to increase or decrease the run process. This is a preferable choice compared to an array as the linked list contains multiple amounts of blocks of memory containing various addresses and allows it to continue running without an initial size being set. For an array, the data does not allow you to alter the data in the block of memory efficiently, nor does it give you an option of having a fixed size. Linked list also allows me to add and remove files updating the node addresses effectively. Furthermore, this allows me to implement different files the network as it is a dynamic data structure meaning if I were to use an array there will be no need to alter the size of memory.

```
Amount of vertex: 6

Node of 0 is weighted by 1 to Node of 4
Node of 0 is weighted by 4 to Node of 1
Node of 1 is weighted by 1 to Node of 5
Node of 1 is weighted by 1 to Node of 3
Node of 1 is weighted by 2 to Node of 2
Node of 2 is weighted by 1 to Node of 4
Node of 2 is weighted by 1 to Node of 3
Node of 3 is weighted by 2 to Node of 4
Node of 4 is weighted by 4 to Node of 5
```

The smallest benchmark which is bridge 1.txt can be seen in the images. The picture presents the total number of nodes, which is six. In addition, showing the vertex connecting with the next in order vertex with the weight, as well as the Ford-Fulkerson Algorithm maximum for that benchmark, which is five. This was accomplished by using the gets for base, point, and author vertices as inputs. The residual graph is used to add the maximum flow, which results in a value of zero. The Residual graph is used while looking for any extra flows that can run, and if there is a path from start to finish, it can add them. The BFS is then run, which includes a few steps to find the maximum path flow and update the residual graph volume, resulting in the edges being reverse after the max flow is returned once added to the pathflows.

```
The max flow of the algorithm is: 0 -- 1 -- 1 -- 5
maxFlow: 1
0 -- 4 -- 4 -- 5
maxFlow: 1
0 -- 1 -- 1 -- 2 -- 2 -- 4 -- 4 -- 5
maxFlow: 1
0 -- 1 -- 1 -- 3 -- 3 -- 4 -- 4 -- 5
maxFlow: 1
0 -- 1 -- 1 -- 2 -- 2 -- 3 -- 3 -- 4 -- 4 -- 5
maxFlow: 1
The Ford Fulkerson flow is 5
Elapsed time = 0. 497 seconds
```

David Sarkodie W1751526 The path flows within the bridge 1.txt file is represented in this illustration. It displays all the links from the beginning node to the end node, as well as the maximum flow for each path it runs. The Ford Fulkerson Max flow from that benchmark is shown after that.

Bridge files total nodes	**Total Elapsed time for each file**	**Ratio Calculation**
6	0.497	
10	0.519	0.519/0.497=1.04
18	0.59	0.59/0.519=1.13
34	0.641	0.641/0.59=1.08
66	0.742	0.742/0.641=1.15
130	1.33	1.33/0.742=1.79
258	4.28	4.28/1.33=3.21
514	16.662	16.662/4.28=3.89
1026	65.913	65.913/16.662=3.95
2050	273.431	273.431/65.913=4.14
Ladder files total nodes		**Ratio Calculation**
6	0.536	
12	0.571	0.571/0.536=1.06
24	0.598	0.598/0.571=1.04
48	0.626	0.626/0.598=1.04
96	0.774	0.774/0.626=1.23
192	1.412	1.412/0.774=1.82
384	4.858	4.858/1.412=3.44
768	18.409	18.409/4.858=3.78
1536	70.796	70.796/18.409=3.84
3072	320.894	320.894/70.796=4.53

The picture on the left displays all the time complexities that were reported for each of the benchmarks that were checked in the project. This was accomplished by using currentTimeMillis() to track the execution of a benchmark. Both the file and bridge network use the Big-O-notation N^2. The explanation for this is that a double for loop is used in the Ford Fulkerson algorithm, which was used to find the pair of nodes that were connected to each other to decide the path from the base to the intended node. While the bridge and ladder files have different data structures, the ratio shift in both benchmarks produces the same returns, as seen in the picture on the left. Furthermore, the output for the ratio shift for the existing biggest benchmarks for both the networks and bridge will always be roughly equivalent to four.