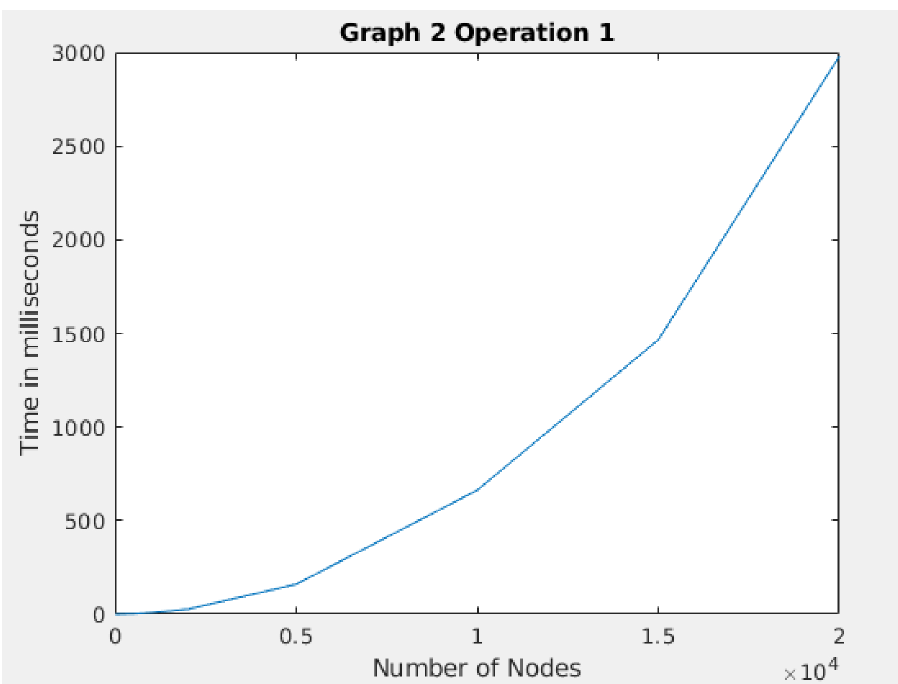
MP6 Performance Evaluation

Heath Gerrald

Part 1) Finding the shortest path on graph 2, I found the following data. This data supports that my implementation does follow a O(n^2) complexity in that every time the nodes N are doubled (besides low values of N) the output is quadrupled. In jobs from 500 to 1000, 1000 to 2000, and 10,000 to 20,000, the output time is 4x the half. To prove that this ratio should be 4, take the examples of 8 and 16. 8^2 = 64, and 16^2 = 256, where 256 / 64 = 4. In further analysis, you can see from my graph below that my data follows a n^2 trend.

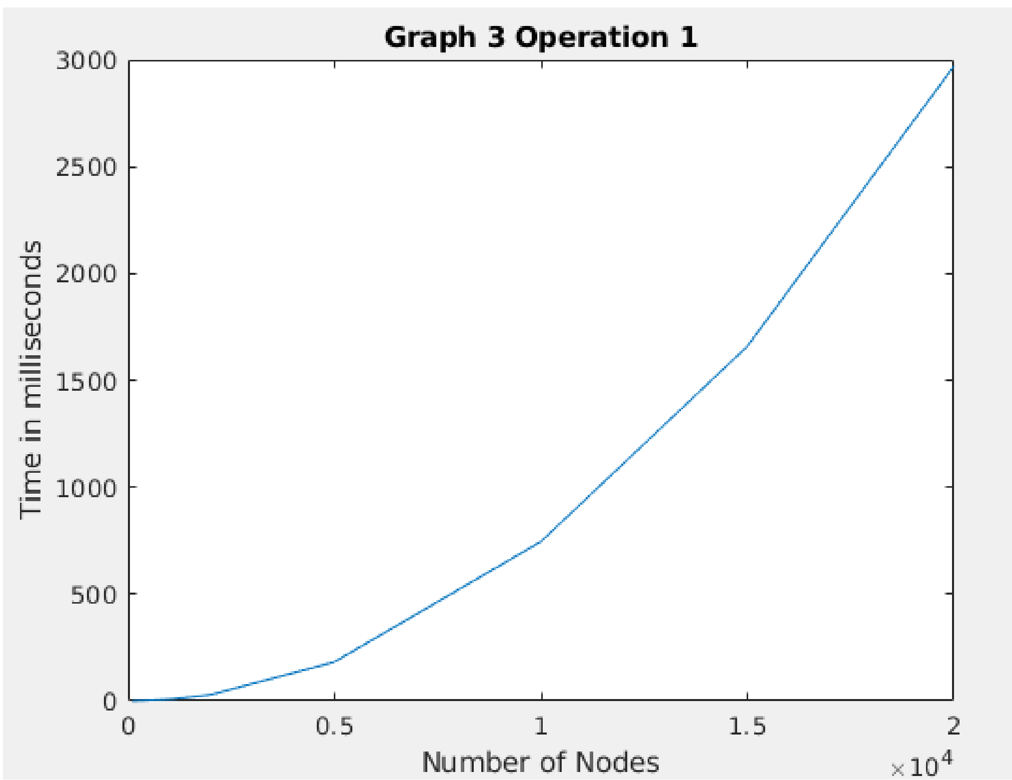
# Nodes	Time (ms)
10	0.067
250	1.54
500	2.045
1000	8.884
2000	26.759
5000	160.174
10000	662.565
15000	1465.592
20000	2979.528



b) Running the same test with the random graph construction and adjacent vertices equal to 20 (-a 20) I collected the following data. These times follow the same pattern as those above, showing that my implementation with random graphs is still O(n^2). The graph illustrating this data is shown below.

* the top input is 100 nodes instead of 10 to keep adjacent vertices similar for all tests

# Nodes	Time (ms)
100	0.382
250	1.666
500	3.719
1000	7.86
2000	28.701
5000	182.977
10000	746.439
15000	1660.748
20000	2970.88



2) My finding network diameter implementation has an O(n^3) complexity. I came to this conclusion since every time the number of nodes are doubled, the output time is 8x the half. With an O(n^2) algorithm, it would be 4x the half. For each jump below where the nodes are doubled, the time typically increases by 8 fold.

# Nodes	Time (ms)
100	31.308
250	141.794
500	884.881
1000	6834.453
2000	56012

3) In experimenting with node density for random graphs, I compared how the adjacent vertices parameter affects the probability that the graph will be connected. My results were as expected – when there are few adjacent vertices there is a less likely chance the graph will be connected.

# Nodes	Adjacent Verts	Seed	Time (ms)	Connected?
100	7	1400	10.836	No
100	7	9494	10.882	Yes
100	7	948859	10.929	No
100	7	2	9.891	No
100	7	10000	9.236	No
100	7	12345	9.378	No
100	7	8765	9.967	No
100	7	2124054	8.861	No
100	7	90093	10.208	No
100	7	412	10.891	Yes

# Nodes	Adjacent Verts	Seed	Time (ms)	Connected?
100	20	40504	13.808	Yes
100	20	616824	12.505	Yes
100	20	34	35.516	Yes
100	20	10001	33.423	Yes
100	20	40059504	14.102	Yes
100	20	213	13.015	Yes
100	20	315	32.626	Yes
100	20	14005	13.503	Yes
100	20	14008	14.048	Yes
100	20	20225	13.273	Yes

4) Below is the data recorded from 5 trials of running multiple link-disjoint paths and each shows there are N-1 paths. In each trial the source node was N-1 and destination node was 0.

# Nodes	# Paths
15	14
22	21
12	11
100	99
1000	999

b) When running the same test but changing the number of adjacent vertices parameter, I discovered that increasing the number of adjacent vertices increases the number of possible paths. This result was as expected.

# Nodes	Adjacent Verts	# of paths
1000	10	2
1000	20	5
1000	50	25
1000	100	49
1000	500	237