CSC2001:ASSIGNMENT 5: GRAPHS

# 1.OOP design

Graph class(which contains the Edge class and Vertex Class) generates graphs given the dataset generated by Graphgen and GraphExperiment class(the main class). It contains methods to generate the shortest path to the node. Dijkstra’s method will be used for the experiment.

Two classes were constructed; Graphgen(Graph Generator) and GraphExperiment. Graphgen takes in the parameter for the Number of vertices, edges and cost limit; It creates and writes to a text file(in data folder as data\_V\_E where V and E are the vertex and edge numbers ) the Nodes, Edges and costs associated with each edge. Nodes are generated randomly and conditions are put in place to remove the possibility of generating repeat nodes. The class also generates a DataX.txt, where X is the number of the dataset containing the number of vertices,edges, vertex operation count, edge operation count, Priority queue counts, and ElogV.

GraphExperiment sets the Number of vertices, edges, and cost limit for the Graphgen class to create data that will be used to generate Graphs using the Graph class. It also calculates ElogV and sums the number of operations that will be written to DataX text file. The class generates 42 experiments and compiles the results per run into a DataX file using different edges and vertices limits. A total of 4 experiments (using a set 42 parameters ie Vertex and Edge number) to test the different graphs that can be made by the Dijkstra shortest path method in Graphs were run.

# Experiment Description

The experiment was carried out by varying the size of V and E , measuring the number of vertex-processing and edge-processing operations for different values of V and E, and additional priority queue operations in the Graph class for Dijkstra’s method to compare operations to the theoretical value.  7 different values of V {10, 20, 30, 40, 50,60,70} and for each value of V, different values of E were used.

For consistency, this experiment was limited to having no repeat edges so that all datasets created had unique edges.

For each pair of values of V and E, the classes described above generate a dataset and measure performance.

Vcount, Ecount and PQ\_count variables were added to Graph’s class Dijkstra method. Vcount was increased by 1 every time a vertex was processed, Ecount was increased by 1 every time an edge was processed, and PQ\_count was increased by 1 every time a priority queue operation occurred.The sum of these operations led to the Operations variable that was compared to Elog(V).

# 2. DATA and RESULTS

2 files of DataX.txt were generated. The following graphs show the results for each run.

.The experiment used the following edge and vertex pairs:

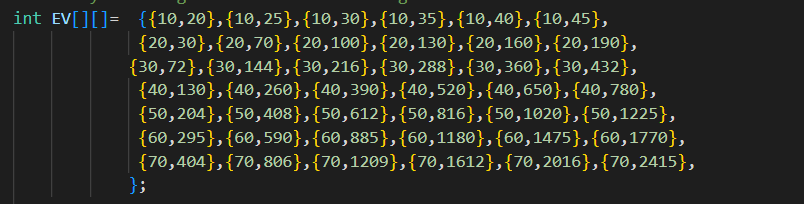


Figure - **6 pairs of V and E for different sets of V: Uniformly distributed Edges**

1st run

Path to datasets and DATA1.txt/

Datasets: data/DATA1

Data1.txt: data/COUNTS/

2. The second run

Datasets: data/DATA2

Data2.txt:data/COUNTS/

## ELOG(V) and Operations

Chart, line chart

Description automatically generated

Figure 2 1st RUN

Chart, line chart

Description automatically generated

Figure 3 2ND RUN

### ADDITIONAL COMPARISONS ARE MADE BELOW:

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated

Figure 4 E and ELOGV Figure priority queue count and ELOGV

Chart, line chart

Description automatically generated

Figure Vertices and ELOGV

## CORRELATION BETWEEN ELOGV AND OPERATIONS

Chart, scatter chart

Description automatically generatedChart, scatter chart

Description automatically generated

Figure 1ST RUN Figure 8 2ND RUN



Figure 1st RUN



Figure 2nd Run

# 3.DISCUSSION OF RESULTS

We expect the theoretical value ElogV to bind the number of operations. The Elog(V) and Operations subplots and the DataX.txt files show that the number of operations is always less than the theoretical value but the two vary almost linearly to each other. They follow the similar trends as E and V vary .

The number of edges and priority queue operations appear to affect the shape of the Operations graph much more than the number of vertex operations which suggest they have a greater bearing…

Elog(V) and Operations are almost perfectly correlated as shown above in Fig7-10 with 0.985 for the first run and 0.982 for the second run of Dijkstra. . The two values are approximately equal to +1 . A +1 correlation indicates a perfect linear relationship.

# 4. CREATIVITY

I used Python’s matplotlib and NumPy to generate graphs to compare E, V, PQ count, and the sum of these parameters with the theoretical bound ELOG(V).

Data1.txt and Data2.txt were uploaded and separated into separate arrays. Column 1 was the number of edges Column2 -was the number of vertices, Column 3 – was the number of edge operations, Column 4 – was the number of vertex operations, Column 5 – was the number of priority queue operations, Column 6 – Elog(V) and Column 7 was the sum of operations.

# 5. GIT USAGE LOG

