COL819: Programming Assignment 2: Implement GHS

March 12, 2021

1 Logistics

- Release date: 12^{th} March 2021
- Due date: To be decided
- Maximum marks: 100
- Can be done in groups of two.
- Submissions on Moodle: Code and report has to submitted.
- Languages: Java (version 8 and above), C, C++, and Python 3.X
- Report must be in LaTeX; use vector graphics for images. Do not save your images in the jpeg, png, or bmp formats. Only save your images in the pdf or svg formats. Draw your diagrams using inkscape and plot the graphs using the matplotlib library.

2 Gallager Humblet Spira (GHS) algorithm [60 marks]

In this assignment, you need to simulate the GHS algorithm to find the minimum spanning tree (MST) of a given graph $G(V, E)$, where V is the set of vertices or nodes in the graph ($|V| = N$), and E denotes the set of undirected edges in the graph ($|E| = M$).

Your task is, given a graph $G(V, E)$, finds its minimum spanning tree (MST). Each node/process should be implemented as a separate thread. You need to create a mechanism to pass messages between the threads. The input format of the graph is as follows:

```
5
(0  ,1 ,  7)
(0  ,2 ,  6)
```

```
(0 ,3, 5)
(0 ,4, 1)
(1 ,2, 4)
(1 ,3, 2)
(1 ,4, 10)
(2 ,3, 9)
(2 ,4, 8)
(3 ,4, 3)
```

The first line is the number of nodes in the graph, and the rest of the lines are edges in the format: (start node, end node, edge weight). Your task is to calculate the MST of this graph using the GHS algorithm and output the final MST in the following format:

```
(0 , 4, 1)
(1 , 3, 2)
(3 , 4, 3)
(1 , 2, 4)
```

The output is the list of edges in the MST sorted by their edge weight in ascending order. Please note the spaces.

Graph Properties:

1. The input graph is connected.
2. The maximum number of nodes (N) in the graph can be $N \leq 400$.
3. The maximum number of edges (M) in the graph can be $M \leq 160000$.
4. All the edges have unique weights. Hence, the graph will have a unique MST.

3 Experiments [20 Marks]

The Message Complexity of the GHS algorithm is $2E + 5Nlog(N)$. You need to do the following experiments for the complexity analysis:

- Run simulations for different sized graphs (Number of nodes: 50, 100, 200, 400). Calculate the total number of messages sent by all the nodes. Compare the total number of messages to $2E + 5Nlog(N)$.
- Repeat the above experiment for a different number of edges in the graph. You need to do two experiments: *sparse graph, dense graph.*
- sparse graph: Construct a random graph by assuming the probability of an edge between two nodes to be 0.2.
- dense graph: Construct a random graph by assuming the probability of an edge between two nodes to be 0.8.

4 Auto-Grading

The assignment will be auto-graded. You need to submit two bash files.

- *compile.sh*: This script should contain all the commands to compile your programs.
- *run.sh*: This script should contain all the commands to run your program. The input-output format is *./run.sh* < *input.txt* >< *output.txt* >, where, *input.txt* contains the graph in the given format and you should write your output in the file *output.txt* in the given format.
- We will run: *./compile.sh; ./run.sh <input.txt> <output.txt>*. There will be no demos for this assignment. You will not get any marks if your code does not run in the given format.

5 Report [20 Marks]

Please show the crucial part of the implementation using code snippets. Please ensure that the report contains the following sections:

1. Introduction to the GHS algorithm.
2. Details of your implementation (using code snippet). Please use the *Listings* package for code snippets and try to avoid inserting images for the code.
3. Plots for the complexity analysis. Explain the trends.

6 General guidance

- Please stick to basic packages during implementation.
- If you are not sure if a particular package is allowed, ask on Piazza.
- We will run MOSS on the submissions. Anyone found with a copied code, either from the internet or from another student, will be dealt with as per the class policy.