COL819: Programming Assignment 2

Task: Implement the GHS algorithm for computing a minimum spanning tree in a distributed fashion.

Download PDF

Logistics

1. Release date: **12 Feb 2020– vial E-mail**
2. Due date:
3. Maximum marks: 100
4. Individual assignment
5. Submissions on Moodle: Code and report has to be submitted.
6. Languages: Java/C++/Scala/Google Go
7. **Report must be in Latex and use vector graphics for images (graphs, if any).**

1 Pre-requisite

1. Prim's algorithm [3, 4] to find the minimum spanning tree.
2. Writing parallel code in a language of your choice.

2 Gallager Humblet Spira (GHS) algorithm

In this assignment you need to simulate the GHS algorithm [2, 5, 1] 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 un-directed edges in the graph ($|E| = M$).

Your task is, given a graph $G(V, E)$ finds its minimum spanning tree (MST). The graph will be given to you in an input file:

Listing 1: Input

```
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)
```

where, 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 print the final MST in this format:

Listing 2: Output

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

This is the list of edges in the MST sorted by their edge weight (ascending order). Please note the spaces.

Graph properties:

1. Graph will be connected.
2. Maximum number of nodes in the graph can be $50 \leq N \leq 100$.
3. Maximum number of edges $200 \leq M \leq 1000$.
4. Edges will have unique weights. Hence, the graph will have a unique MST.
5. A sample test case is provided. Download Input, Output.
6. We will evaluate the algorithm with some hidden test cases. Marks will be based on the correctness of the algorithm.
7. Your code must be able to read the input from a file, and print the final solution on the console. For e.g.

   ```
   <command to run the code> inp
   ```

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

assuming *inp* contains the input shown in Listing 1.

3 Report

Please show the key part of the implementation using code snippets. Also, comment on the run time of the code for the sample input provided (100 nodes). Please ensure that the report contains following sections:

1. Introduction to GHS.
2. Details of your implementation (using code snippet). Please use Listing package for code snippets and try to avoid inserting images.
3. Running time analysis.
4. How to run the code.

Maximum number of pages in the report: 10 (excluding references.)

General guidance

- Please stick to basic packages during implementation.
- If you are not sure if a particular package is allowed, ask on Piazza.
- Grading will be done based on the correctness of the code and the report quality. There might be a demo, if required. So please ensure that the submitted code executed on your machine correctly.
- We will run **MOSS** on the submissions. Anyone found with copied code either the from Internet or from another student, will be dealt with as per the class policy.

References

[1] Assorted algorithms - minimum spanning trees, snapshots. `http://www.cse.iitd.ac.in/~srsarangi/csl860/docs/mst.pdf`. (Accessed on 02/17/2020).

[2] Distributed minimum spanning tree - wikipedia. https://en.wikipedia.org/wiki/Distributed_minimum_spanning_tree#GHS_algorithm. (Accessed on 02/17/2020).

[3] Prims minimum spanning tree (mst) — greedy algo-5 - geeksforgeeks. https://www.geeksforgeeks.org/prims-minimum-spanning-tree-mst-greedy-algo-5/. (Accessed on 02/17/2020).

[4] Chip Martel. The expected complexity of prims minimum spanning tree algorithm. *Inf. Process. Lett.*, 81(4):197201, February 2002.

[5] Artem Mazeev, Alexander Semenov, and Alexey Simonov. A distributed parallel algorithm for minimum spanning tree problem. *CoRR*, abs/1610.04660, 2016.

[6] Gerard Tel. *Introduction to Distributed Algorithms.* Cambridge University Press, USA, 2nd edition, 2001.