COMP9313 2018s1 Project 3 (25 marks)

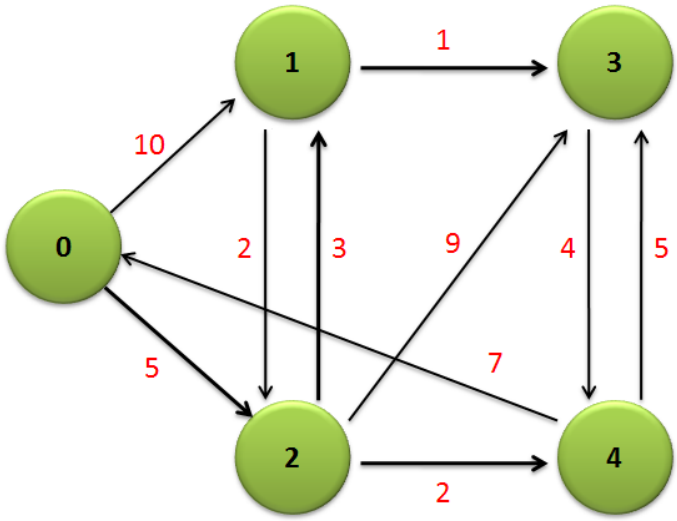
Problem 1 (15 pts, do not use GraphX):

Given a directed graph, compute the average length of the out-going edges for each node, and sort the nodes first according to the average lengths in descending order and then according to the node IDs (numeric value) in ascending order.

Input files:

In the input file, each line is in format of:

"EdgeId FromNodeId ToNodeId Distance".



In the above example, the input is like:

```
0 0 1 10.0
1 0 2 5.0
2 1 2 2.0
3 1 3 1.0
4 2 1 3.0
5 2 3 9.0
6 2 4 2.0
7 3 4 4.0
8 4 0 7.0
9 4 3 5.0
```

This sample file "tiny-graph.txt" can be downloaded at:

https://webcms3.cse.unsw.edu.au/COMP9313/18s1/resources/15769

Output:

Each line in the single output file is in format of "NodeID\tAverage length of out-going edges". The average length is of double precision. Remove the nodes that have no out-going edges. Given the example graph, the output file is like:

```
0\t7.5
4\t6.0
2\t4.66666666666666
3\t4.0
1\t1.5
```

Code format:

Name your scala file as "Problem1.scala", the object as "Problem1", and put it in package "comp9313.ass3". Store the final result in a text file on disk. Your program should take two parameters: the input graph file and the output folder.

Problem 2 (10 pts, use GraphX):

Given a directed graph and a node "t", compute the number of nodes that are reachable from t in the graph.

Input files:

The same format as that in Problem 1.

Output:

A single integer. For example, given the sample tiny graph and node 0, the result is 4, because 0 can reach all the other four nodes in the graph.

Code format:

Name your scala file as "Problem2.scala", the object as "Problem2", and put it in package "comp9313.ass3". Your program should take two parameters: the input graph file and the node ID. Print the result to stdout.

Documentation and code readability

Your source code will be inspected and marked based on readability and ease of understanding. The documentation (comments of the codes) in your source code is also important. Below is an indicative marking scheme:

Result correctness: 90%

Code structure, Readability, and Documentation: 10%

Submission:

Deadline: Sunday 6th May 09:59:59 PM

Log in any CSE server (e.g., williams or wagner), and use the give command below to submit your solutions:

$ give cs9313 assignment3 Problem1.scala Problem2.scala

Or you can submit through:
https://cgi.cse.unsw.edu.au/~give/Student/give.php

If you submit your assignment more than once, the last submission will replace the previous one. To prove successful submission, please take a screenshot as assignment submission instructions show and keep it by yourself.

Late submission penalty

10% reduction of your marks for the 1st day, 30% reduction/day for the following days.

Plagiarism:

The work you submit must be your own work. Submission of work partially or completely derived from any other person or jointly written with any other person is not permitted. The penalties for such an offence may include negative marks, automatic failure of the course and possibly other academic discipline. Assignment submissions will be examined manually.

Relevant scholarship authorities will be informed if students holding scholarships are involved in an incident of plagiarism or other misconduct.

Do not provide or show your assignment work to any other person - apart from the teaching staff of this subject. If you knowingly provide or show your assignment work to another person for any reason, and work derived from it is submitted you may be penalized, even if the work was submitted without your knowledge or consent.