COMP9313 22T2 Project 2 (16 marks)

Problem 1. Top-*k* Most Frequent Co-occurring Term Pairs (8 pts, use Spark RDD):

In this problem, we are still going to use the dataset of Australian news from ABC. Your task is to find out the top-*k* most frequent co-occurring term pairs. The co-occurrence of (w, u) is defined as: u and w appear in the same article headline (i.e., (w, u) and (u, w) are treated equally).

Input files:

In the file, each line is a headline of a news article, in format of "date,term1 term2 ... ... ". The date and texts are separated by a comma, and the terms are separated by the space character. A sample file is like below:

```
20030219,council chief executive fails to secure position
20030219,council welcomes ambulance levy decision
20030219,council welcomes insurance breakthrough
20030219,fed opp to re introduce national insurance
20040501,cowboys survive eels comeback
20040501,cowboys withstand eels fightback
20040502,castro vows cuban socialism to survive bush
20200401,coronanomics things learnt about how coronavirus economy
20200401,coronavirus at home test kits selling in the chinese community
20200401,coronavirus campbell remess streams bear making classes
20201015,coronavirus pacific economy foriegn aid china
20201016,china builds pig apartment blocks to guard against swine flu
```

This sample file “tiny-doc.txt” can be downloaded at:

https://webcms3.cse.unsw.edu.au/COMP9313/22T2/resources/78091

There is also a stop word list stored in the file:

https://webcms3.cse.unsw.edu.au/COMP9313/22T2/resources/78092

Output:

Please get the terms from the dataset as below:

- Ignore the stop words such as “to”, “the”, and “in”.
- Ignore terms starting with non-alphabetical characters, i.e., only consider terms starting with “a” to “z”.

Your Spark program should generate a list of ***k*** key-value pairs ranked in descending order according to the frequencies, where the keys are the pair of terms (the two terms are sorted in alphabetical order and separated by “,”), and the values are the co-occurring frequencies, and keys and values are separated

by "\t". If two pairs have the same frequency, sort them in in alphabetical order when selecting top-*k*.

Given *k* = 3 and the sample dataset, the output is like:

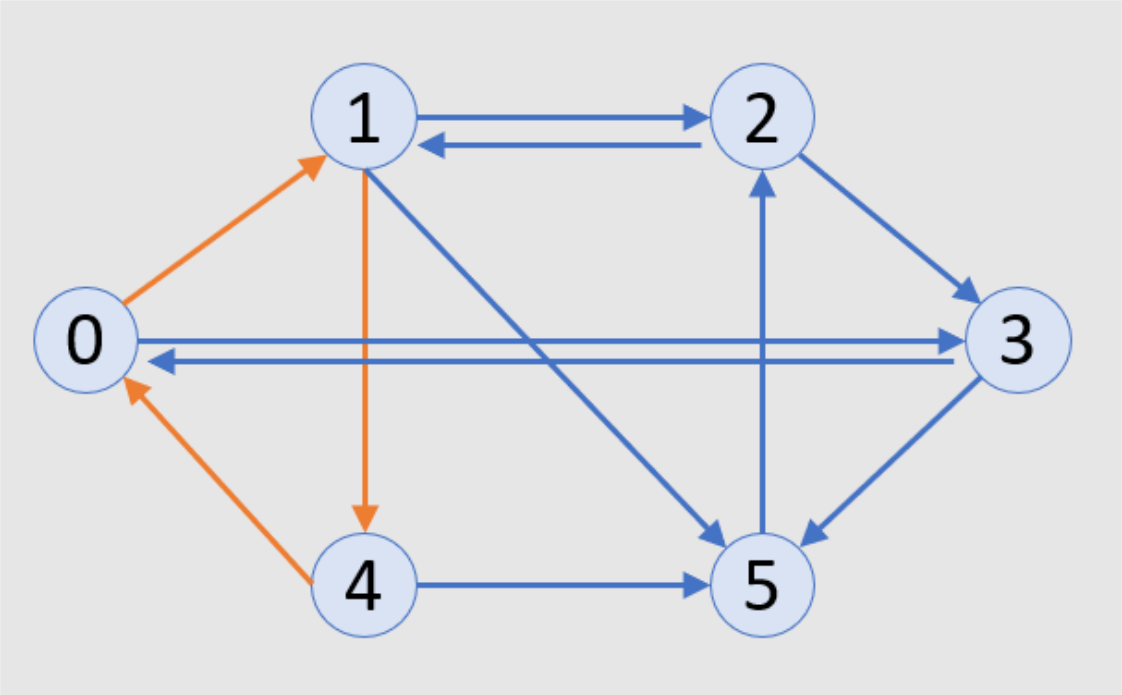
```
coronavirus,economy\t2
council,welcomes\t2
cowboys,eels\t2
```

Code format:

Name the package as "comp9313.proj2", the scala file as "Problem1.scala", and the object as "Problem1". Store the result in a text file on disk. Your program should take four parameters (follow this order): k, the stop words file, the input text file, and the output folder.

Problem 2 (8 pts, use the GraphX pregel operator):

Given a directed graph, for each vertex v, compute the number of vertices that are reachable from v in the graph (including v itself if there is a path starting from v and ending at v). For example, for node 0, the number of vertices that are reachable from 0 is 6, since there exists a path from node 0 to each node in the graph.



Input:

Each line is in format of "EdgeId FromNodeId ToNodeId". Given the above graph, the input file is like below.

```
0 0 1
1 0 3
2 1 2
3 1 4
4 1 5
5 2 1
6 2 3
7 3 0
8 3 5
9 4 0
10 4 5
11 5 2
```

Output:

Each line of the output file contains the vertex id (VertexId) and the number of vertices that are reachable this vertex. Please sort the output according to the ascending order of the vertex ids. For example, given the above sample tiny graph, the output file is like:

```
0:6
1:6
2:6
3:6
4:6
5:6
```

Code format:

Name the package as "comp9313.proj2", the scala file as "Problem2.scala", and the object as "Problem2". Store the result in a text file on disk. Your program should take two parameters (follow this order): the input graph file and the output folder.

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 for both problems:

Result correctness: 6
Efficiency: 1
Code structure, Readability, and Documentation: 1

Submission:

Deadline: Sunday 24th July 11:59:59 PM

Package the two Scala files into a zip file with name “zID_proj2.zip”.

You can submit through Moodle:

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. If you have any problems in submissions, please email to yufan.sheng@unsw.edu.au.

Late submission penalty

5% reduction of your marks for up to 5 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.