COMP9313 21T3 Project 2 (15 marks)

Problem 1. Relative Frequency (8 pts, use Spark RDD):

Given a corpus of documents, compute the relative frequency $f(w_j|w_i)$, i.e.,

$$f(w_j|w_i) = \frac{N(w_i, w_j)}{\sum_{w'} N(w_i, w')}$$

Here, N(., .) indicates the number of times a particular co-occurring term pair is observed in the corpus. In this problem, we consider that w_i and w_j co-occur if w_j appears after w_i in the same line.

Input:

The format of input files is like "pg100.txt". For example, the input could be:

```
The Works of Shakespeare, by William Shakespeare
Language: English
```

Output:

Please get the terms from the documents as below:

- Ignore the letter case, i.e., convert all words to lower case first.
- Ignore terms starting with non-alphabetical characters, i.e., only consider terms starting with "a" to "z".
- Use the following split function to split the documents into terms:

  split("[\\s*$&#/\"'\\,.:;?!\\[\\](){}<>~\\-_]+")

Each line of the output file is in format of "w_i w_j $f(w_j|w_i)$". The results are sorted by the first term (i.e., w_i) in ascending order (alphabetical), then sorted by the relative frequency (i.e., $f(w_j|w_i)$) in descending order, and finally sorted by the second term in ascending order. The relative frequency is of double precision. Given the above example, the output file is like:

```
by shakespeare 0.5
by william 0.5
language english 1.0
of shakespeare 0.5
of by 0.25
of william 0.25
shakespeare by 0.3333333333333333
shakespeare shakespeare 0.3333333333333333
shakespeare william 0.3333333333333333
```

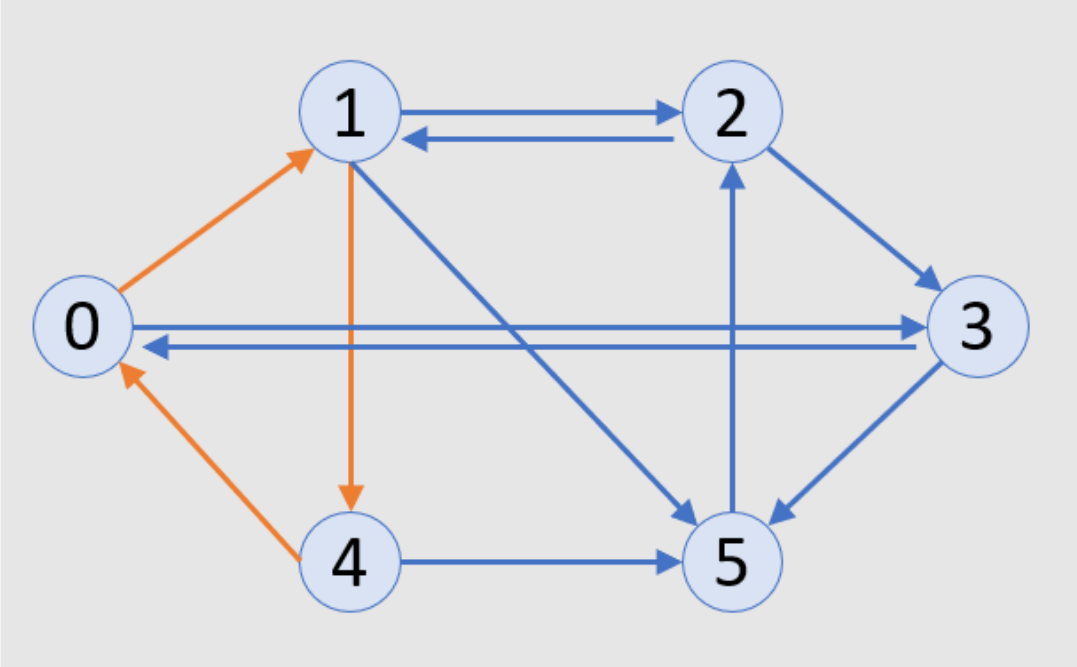
```
the shakespeare 0.3333333333333333
the by 0.16666666666666666
the of 0.16666666666666666
the william 0.16666666666666666
the works 0.16666666666666666
william shakespeare 1.0
works shakespeare 0.4
works by 0.2
works of 0.2
works william 0.2
```

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 two parameters: the input text file and the output folder.

Problem 2 (7 pts, use the GraphX pregel operator):

Given a directed graph, for each vertex, find all the triangles that pass this vertex. A triangle is a cyclic path that passes three different vertices. For example, in the below graph, 0->1->4->0 is a triangle.



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:

For each vertex, display all the triangles that pass it. Each line of the output file contains the vertex id (VertexId) and the triangles (each in format of "VertexId->Vertex1->Vertex2") that pass this vertex, and the format is "VertexId:Triangle_1;Triangle_2;…;Triangle_k;". Please order the lines by VertexId in ascending order numerically. Within each line, sort the triangles on one vertex according to "Vertex1" first and "Vertex2" second in descending order numerically. For example, given the above sample tiny graph, the output file is like:

```
0:0->1->4;
1:1->5->2;1->4->0;
2:2->3->5;2->1->5;
3:3->5->2;
4:4->0->1;
5:5->2->3;5->2->1;
```

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: 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:

Result correctness: 80%
Efficiency: 10%
Code structure, Readability, and Documentation: 10%

Submission:

Deadline: Sunday 7th Nov 11:59:59 PM

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 yu.hao@unsw.edu.au.

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.