Aims

This exercise aims to get you to apply the design patterns you have learned in Chapter 2.2 on MapReduce programming.

Create a folder "Lab3" and put all your codes written in this week's lab in this folder and keep a copy for yourself after you have finished all the problems. Download the input file "pg100.txt" from WebCMS3.

For all problems, using the following codes to tokenize a line of document:

```
import re

words = re.split("[ *$&#/\t\n\f\"\'\\,.:;?!\[\](){}<>~\-_]", line.lower())
```

Problem 1. Compute a Nonsymmetric Term Co-occurrence Matrix Using the "Pair" Approach

The problem is to compute the number of co-occurrences for each pair of terms (w, u) in the document. In this problem, the co-occurrence of (w, u) is defined as: u appears after w in a line of document. This means that the co-occurrence counts of (w, u) and (u, w) are **different!** The task is to use the "pair" approach to solve this problem.

Input is in the format of (line number, line). Output is in the format of ((w, u), co-occurrence count).

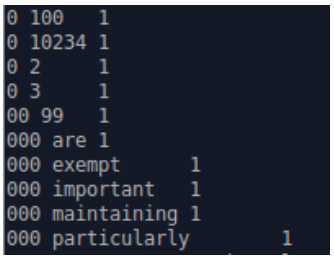
Hadoop Streaming: Create new python scripts CoTermNSPair_mapper.py and CoTermNSPair_reducer.py in the folder Lab3.

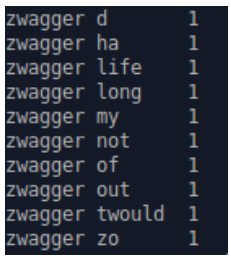
mrjob: Create a new script "mr_CoTermNSPair.py" in the folder Lab3 and solve this problem.

Hints:

1. Refer to the pseudo-code in slide 36 of Chapter 2.2. Note that the condition u is in "NEIGHBORS(w)" means that u appears after w in the same line in this problem.
   For example, give a line "a, b, c, d", for term a, you will generate ((a, b), 1), ((a, c), 1), and ((a, d), 1).
2. What is the map output key? How to store the pair of terms? (A simple method is to concatenate the two terms as a string)
3. How to write the reducer?
4. How about the combiner?

The head and the tail of the result are like:





If your code is correct, you should output 1,161,210 pairs.

Problem 2. Compute a Nonsymmetric Term Co-occurrence Matrix Using the "Stripe" Approach

The problem is the same as Problem 2. The task is to use the “stripe” approach to solve it. Input is in format of (line number, line). Output is in format of ((w, u), co-occurrence count).

Hadoop Streaming: Create new python scripts CoTermNSStripe_mapper.py and CoTermNSStripe_reducer.py in the folder Lab3.

mrjob: Create a new script “mr_CoTermNSStripe.py” in the folder Lab3 and solve this problem.

Hints:

1. Refer to the pseudo-code in slide 39 of Chapter 2.2.
2. You need to use dictionary to formalize the final output string as the map output value.
3. In the reducer, you will receive a list of dictionary objects for the same term w. You need to aggregate them and generate a final “stripe”, and then output the key-value pairs in format of ((w, u), co-occurrence count).
4. The mapper output of Hadoop Streaming is a string that is separated by ‘\t’. The value (second part) of this output should be a string transformed from the dictionary object. How to transform the dictionary to a string that will be processed in the reducer?
5. How to write the combiner? Can you use the reducer as the combiner in this problem? Remember, the input of the combiner is the output of the mapper, and the output of the combiner is the input of the reducer.

If your codes are correct, the results obtained should be the same as obtained in Problem 1.

(Optional) Problem 3. Compute a Symmetric Term Co-occurrence Matrix Using the "Pair" Approach

In this problem, the co-occurrence of (w, u) is defined as: both w and u appear in one line of a document. By this definition, (w, u) and (u, w) are treated as the same. Use the “pair” approach again to solve this problem.

Hadoop Streaming: Create new python scripts CoTermSynPair_mapper.py and CoTermSynPair_reducer.py in the folder Lab3.

mrjob: Create a new script “mr_CoTermSynPair.py” in the folder Lab3 and solve this problem.

Hints:

1. How to modify the codes for Problem 1 slightly to solve this problem?
2. Do you need to change the reducer and combiner?

The generated pairs should be fewer than the nonsymmetric version. The count of (w, u) and (u, w) are merged in the symmetric problem. For example, you will see “a mad 22” and “mad a 21” in the result of Problem 2, and in this problem you will only see “a mad 43” in the output.

If your code is correct, you should output 978,615 pairs.

(Optional) Problem 4. Compute a Symmetric Term Co-occurrence Matrix Using the "Stripe" Approach

The problem is the same as defined in Problem 3. The task is to use the “stripe” approach to solve it.

Input is in format of (line number, line). Output is in format of ((w, u), co-occurrence count).

Hadoop Streaming: Create new python scripts CoTermSynStripe_mapper.py and CoTermSynStripe_reducer.py in the folder Lab3.

mrjob: Create a new script “mr_CoTermSynStripe.py” in the folder Lab3 and solve this problem.

Hints:

1. You only need to modify the mapper. When you get a pair of terms w and u, you need to consider the alphabetical order of w and u. If w < u, you write the information to the stripe for w; otherwise, you need to emit a key-value pair for u, i.e., (u, (w, 1)).
2. Do you need to change the combiner and reducer?

If your codes are correct, the results obtained should be the same as obtained in Problem 3.

Solutions of the Problems

I hope that you can finish all problems by yourself, since the hints are already given. All the source codes will be published in the course homepage on Friday in the same week.