IERG4300/ ESTR4300 Fall 2019 Homework 2

Release date: Oct 2, 2019

Due date: Oct 23, 2019 (Wed) 11:59pm (i.e. mid-nite right before the end of Wed)

The solution will be posted right after the deadline, so NO late homework will be accepted!

Every Student MUST include the following statement, together with his/her signature in the submitted homework.

I declare that the assignment submitted on Elearning system is original except for source material explicitly acknowledged, and that the same or related material has not been previously submitted for another course. I also acknowledge that I am aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website http://www.cuhk.edu.hk/policy/academichonesty/.

Signed (Student_________________________) Date:______________________________

Name_________________________________ SID_______________________________

Submission notice:

- Submit your report in a single PDF document on Elearning

General homework policies:

A student may discuss the problems with others. However, the work a student turns in must be created COMPLETELY by oneself ALONE. A student may not share ANY written work or pictures, nor may one copy answers from any source other than one's own brain.

Each student **MUST LIST** on the homework paper the **name of every person he/she has discussed or worked with**. If the answer includes content from any other source, the student **MUST STATE THE SOURCE**. Failure to do so is cheating and will result in sanctions. Copying answers from someone else is cheating even if one lists their name(s) on the homework.

If there is information you need to solve a problem but the information is not stated in the problem, try to find the data somewhere. If you cannot find it, state what data you need, make a reasonable estimate of its value, and justify any assumptions you make. You will be graded not only on whether your answer is correct, but also on whether you have done an intelligent analysis.

Q0 [20 marks]: Frequent Itemsets: A-priori+SON

Suppose we have a dataset with t item pairs. We want to find frequent pairs with threshold s in terms of percentage. That is, if a pair has a count which is at least $c = t * s$, it is considered to be a frequent pair. Consider the following variation of those algorithms we see in lecture:

1. MapReduce Round 1:
    a. Mapper: Count all the input items and output frequent items, i.e. those items that appear more than s fraction in the input.
    b. Reducer: Collect all the items from the mappers and output a set of candidate items.
2. Collect all outputs from the reducers (there can be more than one reducer) and make a dictionary, denoted by candidate_dict1.
3. MapReduce Round 2:
    a. Mapper:
        i. Load candidate_dict1 as a supplementary file;
        ii.Parse all the input pairs and only store/count a pair if at least one of the item is in candidate_dict1;
        iii.Output the frequent pairs, i.e. those pairs that appear more than s in the input.
    b. Reducer: Collect all the items from the mappers and output a set of candidate pairs.
4. Collect all outputs from the reducers (there can be more than one reducers) and make a dictionary, denoted by candidate_dict2.
5. MapReduce round 3:
    a. Mapper:
        i. Load candidate_dict2 as a supplementary file;
        ii. Parse all the input pairs and only store/count a pair if it is in candidate_dict2;
        iii. Output: <pair, count> tuples. (pair is the key).
    b. Reducer: For each pair, sum up all the counts and output if the total count >= c.

Prove the correctness of this algorithm: **You need to prove and explain whether the following two aspects are true(T) or false(F).**

(a) [10 marks] It will produce no false negative, i.e. all truly frequent pairs are included in the final output.

(b) [10 marks] It may produce some false positives, i.e. some non-frequent pairs may be included as part of its final output

Q1 [80 marks + 20 Bonus marks]: Find frequent item sets

In this problem, we use the Shakespeare data set. The original data set has been pre-processed as follows:

- Apply a sliding window of 40 words on each work. All the 40 words in one window make up a basket.
- Delete duplicate words in one basket, then filter out some common words
- You can download the pre-processed data from the following link: http://mobitec.ie.cuhk.edu.hk/ierg4300Fall2019/homework/shakespeare_basket.zip
- Each line of this input is a tab separated list of words which corresponds to one basket.

The threshold for a frequent pair is defined as s=0.005. The frequency of a pair = *Occurrence of pair (i, j)* / *Total number of baskets*. For Q1(a), (b), (c) and (d), if the number of frequent pairs is larger than 40, please only submit the **Top** 40 pairs (if any). Your result should consist of a frequent pair and its corresponding count.

- You are allowed to use linux command *sort* to post-process your results.

(a) [25 marks] Implement the A-Priori algorithm to find frequent pairs on a single machine

Refer to Lecture 3, Page 30 to implement the A-Priori algorithm. You do not need to use MapReduce Framework for this sub-question. You can run this job on one single AWS machine. Note that dic14.ie.cuhk.edu.hk is only a client for our DIC cluster. Please do NOT run the job on this machine.

(b) [30 marks] Implement the SON algorithm on MapReduce to find frequent pairs

Implement the SON algorithm under the MapReduce framework to find the frequent pairs. Note that your code should be scalable. In other words, your code should allow multiple mappers or reducers in both jobs. You need to implement two MapReduce jobs:

- First MapReduce job should use **A-priori** algorithm to find the candidate pairs, which are frequent in at least one input file.
- Second MapReduce job counts only the candidate frequent pairs.

Tips:

- In the second MapReduce job, each mapper will load all the candidate pairs. You can pass them as a supplementary file.

Streamline and performance comparison:

- Wrap the two MapReduce rounds as single executable by putting those commands you type before in a shell script.
- Compare the overall execution time of (a) and (b) .
- Output the command you use to submit Hadoop job.

You can use the IE Data Intensive Cluster (DIC) or any other Hadoop cluster (e.g., the AWS/Google cluster built in HW#0) in various cloud computing platforms of your choice to do this problem.

(c) [25 marks] SON on MapReduce to find frequent triplets

The threshold for a frequent triplets is defined as s=0.0025. The frequency of a triplet (i, j, k) = <u>*Occurrence of triplet (i, j, k)*</u> / <u>*Total number of baskets*</u>. If the number of frequent triplets is larger than 20, please only submit the top 20 triplets (if any).

Tip:

- In case of memory error, you may need to use multiple mappers/ reducers (*e.g.* 20+).

(d) [20 Bonus marks] Use the PCY algorithm to filter the candidate pairs in the SON algorithm

Implement the SON algorithm under the MapReduce framework. And use the PCY algorithm to filter the candidate pairs in the first MapReduce job. You can use the following Python hash function.

HashFunction=hash(word_1 + word_2) mod 100000

For example, the result of the word pair (‘Monday’, ‘Tuesday’) can be implemented as follows:

HashFunction=hash(‘Monday’ + ‘Tuesday’) % 100000

Streamline and performance comparison:

- Wrap the two MapReduce rounds as single executable by putting those commands you type before in a shell script.
- Compare the overall execution time of (a), (b) and (d).
- Output the command you use to submit Hadoop job.

Part (d) is an optional (bonus) part for IERG4300, but mandatory for ESTR4300.

Submission requirement:

- You need to submit BOTH your code and your result. Please place the relevant code and the result in **a SINGLE PDF** file.