IERG4300 / ESTR4300 Fall 2020 Homework 1

Release date: Sept 23, 2020

Due date: Oct 16, 2020 (Friday) 11:59pm

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 homework via the elearning system

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.

Q1[100 marks + 20 bonus marks]: Community Detection in Online Social Networks

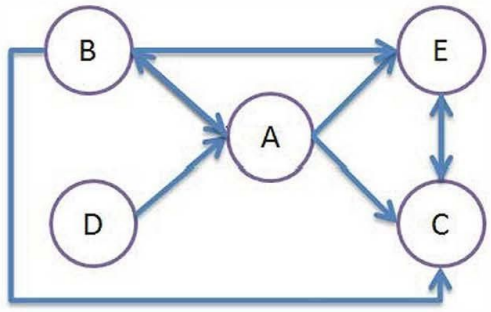
Community detection has drawn lots of attention these years in machine learning field. With the popularity of online social networks, such as Facebook, Twitter, and Sina Weibo, we can get many valuable datasets to develop algorithms. In this homework, you will implement a community detection algorithm for the online social networks. Basically, people belong to the same community when they have high similarity (i.e. sharing many common interests). There are two types of relationship in online social networks. One is symmetric, which means when Alice is Bob's friend, Bob will also be Alice's friend; the other is asymmetric, which means Bob may not be Alice's friend even Alice is Bob's friend. In the second case, there are two roles in the relationship: follower and followee (like the case when using Twitter and Weibo). When Alice follows Bob, Alice is the follower and Bob is the followee.

To detect communities, we need to calculate the similarity between any pair of users. In this homework, similarity is measured by the number of common followees divided by the total number of the two users' followees for the given pair of users. The following figure illustrates the process. The following is the formal definition of similarity, where $\mathbf{out}(A)$ is the set of all the followees of A. If $|\mathbf{out}(A) \cup \mathbf{out}(B)|$ >0, we define

$$\text{Similarity}(A,B) = \frac{|\mathbf{out}(A) \cap \mathbf{out}(B)|}{|\mathbf{out}(A) \cup \mathbf{out}(B)|} \text{ ,.....(*)}$$

where |S| means the cardinality of S.

(**Note**: if $|\mathbf{out}(A) \cup \mathbf{out}(B)| = 0$, we set the similarity to be 0.)



A → B means A follows B

	A	B	C	D	E
A	/	2	1	0	1
B	2	/	1	1	1
C	1	1	/	0	0
D	0	1	0	/	0
E	1	1	0	0	/

Common followees of user pairs

	A	B	C	D	E
A	/	4	3	4	3
B	4	/	3	3	3
C	3	3	/	2	2
D	4	3	2	/	2
E	3	3	2	2	/

Total number of followees of user pairs

	A	B	C	D	E
A	/	0.5	0.33	0	0.33
B	0.5	/	0.33	0.33	0.33
C	0.33	0.33	/	0	0
D	0	0.33	0	/	0
E	0.33	0.33	0	0	/

Pair-wise Similarity

The set of followees of A is {B, C, E} and set of followees of B is {A, C, E}. There are 2 common followees between A and B (i.e. C and E), and the number of the union of their followees is 4. The similarity between A and B is therefore 2/4 = 0.5.

We will provide three datasets with different sizes, generated from FaceBook, Twitter, and Google+ users' relationship. The smaller dataset contains 4K users. The medium one contains around 80K users, and the large one contains 100K users. Each user is represented by its unique ID number. The download links of three datasets are listed in the [1] - [3]. The small dataset is provided to facilitate your initial debugging and testing. The design of your program should be scalable enough to handle all the datasets.

The format of the data file of the above example is as follows:

A B
A D
B A
C A
C B
C E
E A
E B
E C

The output of the community detection is that for *EVERY* user, we want to know the TOP K most similar persons. An example of the output format (K=4) should be (different similar persons separated by space):

A: B C E
B: A C D E
C: A B
D: B
E: A B

Solve the above community detection problem using MapReduce. You can either use the IE DIC or the Hadoop cluster you built for HW#0 to run your MapReduce program(s). You are free to use any programming languages (e.g., Java [4] or Python [5] or C/C++) to implement the required MapReduce components, i.e., mapper(s), reducer(s), etc. (e.g. by leveraging the "Hadoop Streaming" capability [6]).

Again, the design of your program should be scalable enough to handle all three of the datasets.

Submission Requirements:

1. Submit the code and output of the program in one single PDF file.
2. As for part (a), (b), (c) and (e), submit the community detection results of all those users whose IDs share the same last 5 digits with your CUHK student ID. For example, if your student ID is 115500054321, then you need to submit the community members for users with ID = 54321, 154321, 254321, 354321, …, 1154321, ...

a. [20 marks] For EVERY user, recommend the person with the maximal number of common followees in the medium-sized dataset [2]. If multiple people share the same number, randomly pick one of them.

b. [20 marks] Find the top K (K=3) most similar people of EVERY user for the medium-sized dataset in [2]. If multiple people have the same similarity, randomly pick three of them.

(Hint: To facilitate the computation of the similarity measure as defined in (*), you can use the inclusion-exclusion principle $|S \cup T| = |S| + |T| - |S \cap T|$ **)**

c. [30 marks] Besides the number of similar users for a target user, say A, sometimes we also want to know the IDs of the common followees shared between A and its similar users. In our example, for User A, if B is the similar user (TopK in Q1.b) to A, the desirable output should be:

A: B, {C, E}, check_sum for the IDs of the common_followees……(F1)

d. [30 marks] Run part (a) for the **medium** dataset multiple times while modifying the number of mappers and reducers for your MapReduce job(s) each time. You need to examine and report the performance of your program for at least 4 different runs. Each run should use a different combination of number of mappers and reducers. For each run, performance statistics to be reported should include: (i) the time consumed by the entire MapReduce job(s) and the maximum, minimum and average time consumed by (ii) mapper and reducer tasks and (iii) Tabulate the time consumption for each MapReduce job and its tasks. One example is given in the following table. Explain your observations.

Mapper num	Reducer num	Max mapper time	Min mapper time	Avg mapper time	Max reducer time	Min reducer time	Avg reducer time	Total job
3	2	60s	40s	50s	60s	40s	50s	2.5 min

e. [Bonus 20 marks] Find the TOP 3 (=K) most similar people and the list of common followees for each user in the large dataset in [3] using the format of Q1(b). (Hints: To reduce memory consumption of your programme, you may consider to use the composite key design pattern and secondary sorting techniques as discussed in [7] and [8]).

Part (e) is a bonus question for IERG4300 but required for ESTR 4300.

Hints:

1. For this homework, since all user IDs are integers, we will simply take the sum of the IDs of the common followees between A and the similar user as the check_sum.
2. For EVERY user in the medium-sized dataset in [2], find its top K (K=3) similar users, and for each one of these K similar users, output a line following the format of (F1).
3. You can use multiple (i.e. a series of different) MapReduce jobs to handle the above community detection problem. However, it may be more difficult to just use one MapReduce program to get the final results of the above problem due to the memory exhausting problems.
4. For students who cannot setup the single node Hadoop cluster in HW#0, please contact the TAs. You can either choose to set up a single node hadoop cluster with TAs' help or you can apply an IE DIC Cluster account to run MapReduce program. Hadoop has already been installed in IE DIC cluster. Please note that the Hadoop in IE DIC cluster is based on Hortonworks, and it is a little different from the standard Hadoop system. For more information, please refer to [9]. TAs will provide more details in the tutorial.
5. If you use Java, you can specify the number of mappers with the following code: `job.setNumMapTasks(20).` If you use Hadoop streaming with Python, you can specify it via the following command option: `-D mapred.map.tasks=20`. If this does not work,

you may need to modify the split size in the `$hadoop/etc/hadoop/mapred-site.xml: mapred.min.split.size =268435456.` Refer to [10] for more information.

6. As for the large dataset, you may want to set mapreduce.map.output.compress=true to compress the intermediate results, in case you don't have enough local hard disk space (to hold the intermediate tuples).
7. The large dataset may take a long time to process even if everything is correct. **You should start doing this homework as early as possible**.

References:

[1] Small scale dataset
https://www.dropbox.com/s/ns771cy27fpjthj/facebook_combined.txt.gz?dl=0
[2] Medium scale dataset
https://www.dropbox.com/s/g1nthvj98iaikly/twitter_combined.txt.gz?dl=0
[3] Large scale dataset
https://www.dropbox.com/s/cbf4r2at8dmd9ms/gplus_combined.txt.gz?dl=0
[4] Write a Hadoop program in Java
https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html
[5] Write a Hadoop program in Python2
http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/
[6] Hadoop Streaming
https://hadoop.apache.org/docs/r2.9.2/hadoop-streaming/HadoopStreaming.html
[7] Composite Key
http://tutorials.techmytalk.com/2014/11/14/mapreduce-composite-key-operation-part2/
[8] Secondary Sort
http://codingjunkie.net/secondary-sort/
[9] Hortonworks
https://hortonworks.com/
[10] How many Mappers and Reducers?
https://cwiki.apache.org/confluence/display/HADOOP2/HowManyMapsAndReduces