CS24200: Project 3

Due date: Monday November 25, 11:59 pm EST

Unless otherwise specified, use python to complete this assignment. Submit a PDF with both the code (python+unix commands) that you used for analysis and your answers to the questions below. Your homework must be typed.

Download the Twitter CIKM 2010 dataset from:
`https://archive.org/details/twitter_cikm_2010`
This dataset is a collection of scraped public twitter updates used to study the geolocation data related to twittering. You will use the tab separated files `test_set_tweets.txt` and `training_set_tweets.txt`, where each line is of the form: UserID, TweetID, Tweet, CreatedAt.
Note: Some lines do not follow the above pattern. This usually is due to a return/enter midway or at the end of the Tweet. Such lines should be treated as legitimate lines themselves, with fewer tabs (possibly none) in them.
Use the first 500,000 lines of the file for your analysis below.

Download the Twitter friend network dataset from:
`http://socialcomputing.asu.edu/datasets/Twitter`
This dataset is a graph dataset that represents the follower relationships among 11 million Twitter users. You will use the comma separated file `edges.csv`, where each line is of the form: UserID, FollowingUserID.
Use the first 500,000 lines of the file for your analysis below.

For the Unix commands to be used in this homework, the following links might be useful:
grep, sed, awk, sort

In this assignment, you will compute aggregate statistics of the data using mapreduce and command line processing.

1 Finding Trends (15 pts)

1.1

Parse the `test_set_tweets.txt` to (1) extract hashtags (i.e., anything starting with "#"), (2) convert to lower case and remove all non-alpha numeric characters, (3) count the occurrences of each hashtag, and (4) return the top 10 hashtags (i.e., with largest number of occurrences).

(a) Write a map reduce approach in python to accomplish this task. Make sure to include a mapper function, reducer function, and execute function as we discussed in class.

(b) Report the wall clock runtime of your program when applied to the first 500,000 lines of `test_set_tweets.txt`. Use the function `time()` from the python `time` package.

(c) Report the top ten hashtags with their counts.

(d) Write a command line program to accomplish the same task, using e.g., grep, sed, awk, sort.

(e) Run your command line program in a shell script to record the wall clock runtime with e.g., the unix command `time`.

(f) Discuss how the runtimes compare between the two approaches.

1.2

Use the Unix CAT command (or otherwise) to join the first 500,000 lines from the file `test_set_tweets.txt` and the first 250,000 lines from the file `training_set_tweets.txt` into one file, say `tweets.txt`. Parse this file `tweets.txt` to (1) extract usernames (i.e., anything starting with "@"), (2) count the occurrences of each username, and (3) return the top 10 usernames (i.e., with largest number of occurrences).

(1) Repeat **1.1 (a) - (f)** for this task above.

(2) Within the file `tweets.txt`, find the number of tweets that have at least two hashtags. Again, repeat **1.1 (a) - (f)** for this.

2 Finding Reciprocal Followers (15 pts)

Process the follower network data to determine reciprocal following relationships, i.e., pairs of users that mutually follow each other.

(a) Write a map reduce approach in python to accomplish this task. Make sure to include a mapper function, reducer function, and execute function as we discussed in class.

(b) Report the wall clock runtime of your program when applied to the first 500,000 lines of `edges.csv`. Use the function `time()` from the python `time` package.

(c) Output the results in a text file to use in the next question. This will just be a subset of the original `edges.csv` file. Report the difference in size between the two versions of the graph with respect to number of unique nodes, and number of edges.

(d) Write a command line program to accomplish the same task, using e.g., awk, sort, join.

(e) Run your command line program in a shell scrip to record the wall clock runtime with e.g., the unix command `time`.

(f) Discuss how the runtimes compare between the two approaches.

3 Finding Friends of Friends (15 pts)

Use the symmetric follower graph from the question above. For each pair of friends (i.e., pair of linked users), find the number of friends they have in common.

(a) Write a map reduce approach in python to accomplish this task. Make sure to include a mapper function, reducer function, and execute function as we discussed in class. *Note*: you will probably need two map/reduce functions to accomplish this task: one to identify the friends of each user, and another to find the friends they have in common.

(b) Report the top ten pairs and how many friends they have in common.