Homework 1

COL 761

- You need to do the homework in your already formed team of 3
- Due: **18th October 11:59PM.**
- **30%** penalty for each late day
- Your code must compile and execute on HPC. You will get 0 for compilation or execution errors
- No multi-threading or parallelization is allowed

---

1. Create a GitHub ID for your team. Share the GitHub repo with us here. All homework submissions should be uploaded to this GitHub repo. Clearly mention the name and roll no. of each team member in the group on the GitHub Repo [5 points].

2. This question is on frequent itemset mining. Implement Apriori Algorithm and FP-tree to mine frequent itemsets. Apply it on the Dataset: http://fimi.uantwerpen.be/data/webdocs.dat.gz. You may assume all items are integers.
    a. Please name your bash file RollNo.sh. For example, if MCS162913 is your roll number, your file should be named MCS162913.sh. Executing the command "./RollNo.sh retail.dat X -apriori <filename>" should generate a file filename.txt containing the frequent itemsets at >=X% support threshold with the apriori algorithm. Similarly, executing the command "./RollNo.sh retail.dat X -fptree <filename>" should generate a file filename.txt containing the frequent itemsets using FPtree algorithm. Notice that X is in percentage and not the absolute count. Your implementations must ensure that the transactions are not loaded into main memory. This means, that it is not allowed to parse the complete input data and save it into an array or similar data structure. However, the frequent patterns and candidate sets can be stored in memory. *(20+20=40 points)*

       filename.txt should strictly follow the following format.

        i. Each frequent itemset must be on a new line.

        ii. The items must be space separated and in ascending order of ASCII code.

       Your grade will be (F-score apriori)*20 + (F-score fptree)*20.

    b. Compare the performance of apriori algorithm with FP-tree. The FP-tree implementation and Apriori implementation must be in the same language. Executing the command "./RollNo.sh retail.dat -plot" should generate a plot using matplotlib where the x axis varies the support threshold and y axis contains the corresponding running times. It should plot the running times of FP-tree and Apriori algorithms at support thresholds of 1%, 5%, 10%, 25%, 50%, and 90%. Explain the results that you observe. (20 points)

c. Efficiency Competition: We will have a competition among all submitted implementations of the FP-tree. The fastest would get full points. If your algorithm is X% slower than the fastest, then you would get **X% less than** full points. You would be in this competition only if you get full points in part (a), i.e., you have the correct implementation of the FP-tree algorithm. (20 points)

Bash scripts you need to provide:

- compile.sh that compiles your code with respect to all implementations. Specifically running ./compile.sh in your submission folder should create all the binaries that you require. Any optimization flags like O3 for g++ should be included here itself
- RollNo.sh as specified earlier
- install.sh that should execute cloning of your team's repository in the current directory followed by executing bash commands to load all the required HPC modules. Inside the repository we should be able to locate EntryNo-Assgn1.zip corresponding to your Homework 1 submission. **Note this script will be run as source install.sh (to make use of HPC module load alias).**

Submission Instructions:

- Add **col761-2020** as a collaborator for your private repository
- Make sure to fill this form https://forms.gle/D192iASZnRdt4D6r8
- There should be only **one** submission per group
- Upload EntryNo-Assgn1.zip file to the root directory of your GitHub repository. This entry should be of any one of your team. Ex. MCS162913-Assgn1.zip. On unzipping it should produce one folder. The folder should have the same name as zip file. This folder should contain all the source files and all the bash scripts. In addition, it should all contain a README.txt explaining all the files you bundled, explanation of part b and entry numbers and names of **all team** members.
- <Rollno>.sh is the main script that will be used in part a, b, and c
- **You also need to submit install.sh script on Moodle (Do not zip it), one per team**
- Since your submissions will be auto graded, it is essential to ensure your submissions conform to format specified
- Later of timestamp of your last pushed commit and Moodle submission of install.sh will be treated as your submission time

Compiler Specification:

- GCC version 7.1.0
- Java version 1.8
- Python3 version 3.6.5
- Python2 version 2.7.13

Plagiarism Policy:

Do not copy code from your friend or from the internet. We have previous year's submissions and downloaded all available libraries of Apriori algorithm and FP-tree from the web and all submitted codes will be checked against these as well as those submitted in this homework. Any plagiarized code will result in an F grade for the course straight-away.