School of Computing

CS4225/CS5425 Big Data Systems for Data Science

Assignment 1: Introduction and Hadoop

Outline

- Description of Tasks 0&1
- Submission requirements

Task Overview

- Motivation
  - Text and documents are big data.
  - Text and document processing is fundamental for many Web applications.

- The assignment consists of an example program (Task 0: to help you test your setup and give you an example of a working program, no need to submit it) and the actual task to submit (Task 1):
  - Task 0: A simple word count program for testing your setup.
  - Task 1: Given two textual files, count the number of words that are common.

Task 0

- This is a simple test program to test your setup and get accustomed to logging in and working with the cluster
- The code is already given to you and will not be graded. You can use it as an example of what a working MapReduce program looks like.
- To submit this, follow the instructions in the Student Guide sections 3-5. If the test passes and you can successfully submit, it means everything has been set up correctly.

Task 1

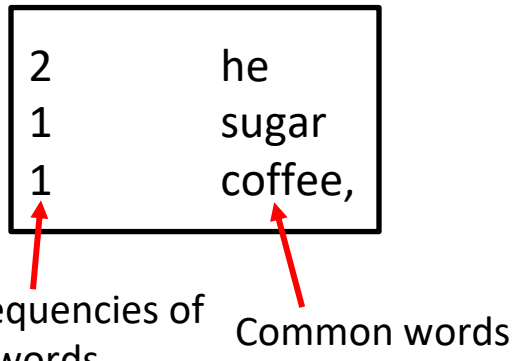
- Motivation
  - We choose **CommonWords** as our task because it is representative, and it is based on **WordCount**.

Task 1

- Problem definition
  - Given TWO textual files, for each common word between the two files, find the *smaller* number of times that it appears between the two files. Output the top 20 common words with highest such frequency (For words with the same frequency, there's no special requirement for the output order).
  - Example: if the word "John" appears 5 times in the 1st file and 3 times in the 2nd file, the smaller number of times is 3
- Requirements
  - Split the input text with "(space)\t\n\r\f". Any other tokens like ",.:`" will be regarded as a part of the words
  - Remove stop-words as given in Stopwords.txt, such as "a", "the", "that", "of", … (case sensitive)
  - Sort the common words in descending order of the smaller number of occurrences in the two files.
  - In general, words with different case or different non-whitespace punctuation are considered different words

A Running Example

- Stopwords
  - as, had
- File 1
  - he put some sugar into his coffee, ~~as~~ he always did.
- File 2
  - he ~~had~~ sugar in his coffee, though he is diabetic and he suffer.
- Output:



Task 1

- TopkCommonWords <input1> <input2> <input3> <output>
- Input data are provided
  - the following two source files:
    - Task1-input1.txt as <input1>
    - Task1-input2.txt as <input2>
  - the stop-word file:
    - Stopwords.txt as <input3>
- Output
  - Output directory is specified in <output>
  - Top-20 output of the result using the data files listed above (you only need to extract these 20 output from the sorted output), with each line:
    - **Freq**\t**word**\n, where \t is tab and \n creates a new line.
- An example command can be:
  - hadoop jar cm.jar TopkCommonWords commonwords/input/task1-input1.txt commonwords/input/task1-input2.txt commonwords/input/stopwords.txt commonwords/cm_output/
  - Note that the ./compile_run script already contains this command. So you can simply test your code using ./compile_run (on the SoC cluster)
- You can directly parse the parameters to get the path, and load the file from the path in your codes. In local debugging, you should set system path as parameters. In clusters, you only need to run "compile_run". The script will automatically upload the files to HDFS and set HDFS path as parameters.

Submission Requirements

- Deadline: Oct 2, 2021, Sun 11:59pm
- Task 0: You can optionally run ./submit on the SoC cluster to test your setup, but it will not be graded
- Task 1: Run ./submit on the SoC cluster to submit your codes
  - When submitting, make sure you input your matriculation number (i.e. starting with capital A), not your email address
  - If the ./compile_run script says that the tests passed, and the ./submit script says that you have successfully submitted, then you have completed the assignment, no need to submit anything else.

Marking

- The assignment contains a public dataset 'data/' and expected output 'answer.txt'. If your codes are correct, your output should be the same as 'answer.txt'. Different orders will also be considered as correct in marking.
- All the codes will be automatically compiled and marked by similar scripts as 'compile_run' on a private test dataset
  - The private test dataset is very similar to the test data given to you. If your program gives the correct output on the given test data, it is very likely to give the correct output on the private test data as well.
- So, ensure your codes can be compiled by the script in your package.
- All submitted codes will be auto-checked for plagiarism. Do NOT copy others' codes or share your codes with others.

Marking Schemes

- Total: 25% of final mark.

Notice

- We have zero-tolerance on plagiarism.
- Do not “Copy and paste” from others.
- Do not share your code with others.

Feedbacks are Welcome

- Email: Xie Yuxi, xieyuxi@u.nus.edu
- Or, post your questions in the LumiNUS forum (preferred).
- You may find something useful from FAQ of last semester, which can be found in the attached student guide. Note: we have change the format of our assignment this year (thus this FAQ is just for your reference).