Lab Course: Distributed Data Analytics
Exercise Sheet 4

Prof. Dr. Dr. Schmidt-Thieme, Daniela Thyssens
Information Systems and Machine Learning Lab
University of Hildesheim
Submission deadline: Saturday June 4, 23:59PM (on LearnWeb, course code: 3116)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit two items a) a zipped file containing python scripts and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission needs to be made before the deadline, only through learnweb.
4. Unless explicitly stated, you are not allowed to use scikit, sklearn or any other library for solving any part. All implementations must be done by yourself.

Exercise 1: Preparing your Hadoop infrastructure

Exercise 1 is an installation guide rather than a graded exercise. You are not required to submit a "solution" for Exercise 1 in order to earn points. However demonstrating that your installation was successful with a solution to the verification exercise 1.3 would be appreciated. You must provide solutions for Exercise 2 on-wards.

Exercise 1.1: Setting up a Hadoop infrastructure

In this exercise, you are asked to set up and configure a single-node Hadoop installation so that you can quickly perform simple operations using Hadoop MapReduce and the Hadoop Distributed File System (HDFS). There are three possible installation modes that you can start your Hadoop cluster:

- Local (Standalone) mode.
- Pseudo-Distributed mode.
- Fully-Distributed mode.

In this exercise sheet, you are asked to install Hadoop in Pseudo-Distributed mode on your own machine.

If you are using a linux distro, e.g. Ubuntu, you can refer to
`http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html#Pseudo-Distributed_Operation`.

For Windows users, you can refer to `http://hadooponwindows10.blogspot.de/`.

In some cases, depending on your physical computer, setting up a Hadoop environment might be a headache duty. The problems might emerge by wrong YARN and MapReduce parameters configurations. Another detailed installation guide and parameters setting can be found here:
`http://www.alexjf.net/blog/distributed-systems/hadoop-yarn-installation-definitive-guide/`

A note for Windows users if you follow the installation instructions above. You might usually set up your Windows log-in account as (`first name`) (`an empty space`) and (`last name`). Then the space between your first and last names might cause the `error could not find or load main class` exception. One possible solution for this problem is that you create a new Windows account without space in account's

name, e.g. lab; grant it as a computer administrator; then re-do the same installation instructions.

You might also install `MinGW-w64` for a complete runtime environment for gcc to support binaries native to Windows. You can find it here `https://sourceforge.net/projects/mingw-w64/`.

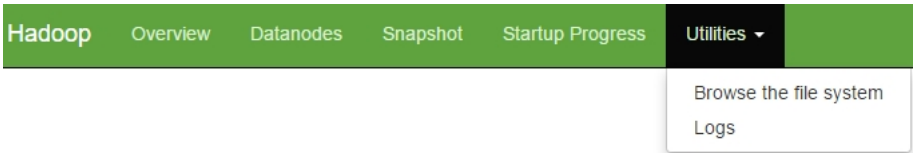
After finishing setting up a Hadoop development environment, let's check if the installation is correct.

1. For Ubuntu users, run `hadoop` with `sudo` right, or one can set `sudo su -hadoop` once. For Windows users, run `cmd` Command Prompt as administrator.
2. Initialize the HDFS file system by the command: `hadoop namenode -format`. You might need to run this command once.
3. After formatting the HDFS, start the distributed file system by the command `start-all.sh` or `start-all.cmd` in Ubuntu or Windows OS respectively. These files can be found in `hadoop/sbin/` directory.
4. Check the Hadoop User Interface in the localhost at the default URL address `http://localhost:50070/`
5. Verify all applications of a cluster in the localhost at the default URL address `http://localhost:8088/`
6. Let's try your first command in the fs shell by `hadoop fs -help`
7. Shut down the HDFS by the command `stop-dfs.sh` or `stop-all.cmd`.

Exercise 1.2: Basic Hadoop operations

In this exercise, you are going to get familiar with Hadoop by playing around some basic Hadoop commands. Report the results that you get to your submitted .pdf to answer this exercise. All Hadoop commands are invoked by the `$HADOOP_HOME/bin/hadoop` command. Running the `hadoop` without any arguments prints the description for all commands.

1. Check Hadoop version: `hadoop version`.
2. List files in HDFS: `hadoop fs -ls /`
3. Create a `hadoopdemo` directory: `hadoop fs -mkdir /hadoopdemo`
4. Create several sub-directories nested in `hadoopdemo`, e.g. `text_files`, `raw_data`
5. Transfer and store a data file from local systems to Hadoop:
   `hadoop fs -put file.txt /hadoopdemo/text_files`
6. Check directories and files using Hadoop User Interface
   `http://localhost:50070` > Utilities > Browse the file system.



7. Remove the sub-directory `hadoop fs -rm -r /hadoopdemo/text_files`
8. Change the content of `file.txt` in the local system and overwrite it in Hadoop
   `hadoop fs -put -f file.txt /hadoopdemo/text_files`
9. Read the content of the file: `hadoop fs -cat /hadoopdemo/text_files/file.txt`

10. Try other useful commands
    ```
    hadoop fs -copyFromLocal <localDir> <hdfsDir>
    hadoop fs -copyToLocal <localDir> <hdfsDir>
    hadoop fs -count <path>
    hadoop fs -du -h <path>
    hadoop fs -stat <path>
    ...
    ```

More Hadoop commands can be found at `https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/CommandsManual.html`.

Exercise 1.3: WordCount MapReduce example

Before going further to the exercise, let's learn some useful commands to interact with MapReduce jobs. General usage: `hadoop job [GENERIC_OPTIONS]`.

1. List generic options available in a Hadoop job: `hadoop job`
2. Check the status of job: `hadoop job -status <JOB-ID>`
3. List all running jobs: `hadoop job -list`
4. Check the history of job output-dir: `hadoop job -history <DIR>`
5. Kill a job: `hadoop job -kill <JOB-ID>`
6. To monitor your cluster and your jobs, you can also check the web interfaces of the Hadoop components instead of using the command line. The default address: `http://localhost:8088/cluster/apps`

In order to practically understand how data flows through a `map` and `reduce` computational pipeline, you are going to implement a WordCount example. WordCount is a simple application that counts the number of occurrences of each word in a given input set. More specifically, the input key is a file and the input value is a string, e.g. the file's content, while the output key is a word and the equivalent output value is an integer. Further descriptions about the WordCount application can be found here `http://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html`.

In this exercise, you are going to run a version of WordCount program that is already shipped with Hadoop installation. The successful execution of the program indicates that your Hadoop infrastructure is configured correctly and ready to use. The text dataset used for this exercise can be found here: `https://www.mirrorservice.org/sites/ftp.ibiblio.org/pub/docs/books/gutenberg/1/0/101/101.txt`.

In order to prove that you have done this exercise, you are asked to describe your solution step-by-step, e.g. command lines that you use, pictures of output, pictures of Hadoop User Interface, from the first command to last one. Please include the screenshot of the final console output for this step in your report.

Exercise 2: Analysis of Airport efficiency with Map Reduce (10 points)

In this exercise you will download data from Bureau of Transportation Statistics' homepage[1]. On the Bureau' homepage you will download data by selecting following fields:

- Filter geography: all
- Filter year: 2017

[1] `https://www.transtats.bts.gov/DL_SelectFields.aspx?gnoyr_VQ=FGJ&QO_fu146_anzr=b0-gvzr`

- Filter period: January
- Time period: FlightDate
- Airline: Reporting Airline, Flight Number Reporting Airline
- Origin: Origin
- Destination: Dest
- Departure Performance: DepTime, DepDelay
- Arrival Performance: ArrTime, ArrDelay

You will get a CSV file containing 450017 lines. An example of a line is as follows
`1/1/2017 12:00:00 AM,AA,307,DEN,PHX,1135,-10,1328,-17`

In this exercise, you are going to write a python code using **Hadoop MapReduce** to accomplish several requirements:

1. Computing the maximum, minimum, and average departure delay for each airport.[Hint: you are required to find max, min and avg in a single map reduce job]
2. Computing a ranking list that contains top 10 airports by their average Arrival delay.
3. What are your `mapper.py` and `reduce.py` solutions?
4. Describe step-by-step how you apply them and the outputs during this procedure.

Exercise 3: Analysis of Movie dataset using Map and Reduce (10 points)

For this exercise you will use movielens10m dataset available at `http://files.grouplens.org/datasets/movielens/ml-10m-README.html`. The movielens10m dataset consists of 10 million rating entries by users. There are two main files required to solve this exercise 1) rating.dat and 2) movie.dat. The rating.dat file contains userId, movieId and ratings (on scale of 1 to 5). The movie.dat file contains information about movies i.e. movieId, Title and Genres.
In this exercise, you are going to write a python code using **Hadoop MapReduce** to accomplish several requirements, note as your dataset is large you also need to perform some performance analysis i.e. how many mappers and reducers you used, what is the performance increase by varying number of mappers and reducers. Please also note that although you may have very few physical cores i.e. 2 or 4 or 8, but you can still experiment with a large number of mappers and reducers. **Plot the execution time vs the number of mappers $\times$ reducers for each task below.**

1. Find the movie title which has the maximum average rating?
2. Find the user who has assign lowest average rating among all the users who rated more than 40 times?
3. Find the highest average rated movie genre? [Hint: you may need to merge/combine movie.dat and rating.dat in a preprocessing step.]