BIGDATA TEAM

HW #02: MapReduce

Deadline: see in the table with grades

Homework author: unknown

1. Homework Description and Grading Criteria

In this homework you have to solve one Programming Assignment on Hadoop Streaming.

Optimal MapReduce solution will:

- contain as least Hadoop Jobs as possible;
- likely contain Combiner to speed-up calculations;
- have more than 1 reducers to spread the load for interim Hadoop Jobs. It is allowed to use 1 reducer for the last Hadoop job where you sort results.

Before you start implementing your solution:

- Follow Ch. 3 Job Chaining and Ch. 4 General Recommendations.

Criterias:

- **60%** - solution is correct
- **20%** - readability and maintainability of your code (see “Clean Code” and Google Python Style Guide)
- **20%** - efficiency of your solution

Bonuses and discounts:

- **100%** for plagiarism (to all involved learners)
- **30%** for sending solution 1 week after deadline
- **10%** for submission rules violation
- **5%** for every extra submission to the Grader

2. Programming Assignment[1]: Popular Bigrams.

Find the most popular bigrams (pair of words) in Wikipedia sample dataset.

Programming Assignment description:

- transform all words into lowercase;
- remove all punctuation marks;
- sort bigrams by popularity, i.e. pairs (bigram, counts) should be sorted in descending order by counts;

[1] Internal ID - 116

- if you have several pairs (bigram, counts) with the same value of “counts”, sort them lexicographically by “bigram” (e.g. "aaa" goes before "bbb").

Input Data

Wikipedia:

- Path: **/data/wiki/en_articles_part**
- File format: text;
- Line format: article_ID <tab> article_content;

Output Data

HDFS:

word_1 **<space>** word_2 <tab> amount_of_bigram_occurences_in_dataset

STDOUT:

print TOP-10 bigrams

Output example:

```
and the 3495
on the 3326
by the 3250
```

3. Job Chaining.

MapReduce Job Chaining example:

- in short: "`( ... && ... ) || echo 'smth' "`

run.sh:

```
#!/usr/bin/env bash
set -x

HADOOP_STREAMING_JAR=/opt/cloudera/parcels/CDH/lib/hadoop-mapreduce/had
oop-streaming.jar
OUT_DIR="streaming_wc_result"
NUM_REDUCERS=8

hdfs dfs -rm -r -skipTrash ${OUT_DIR}* > /dev/null
```

```
# Wordcount
( yarn jar $HADOOP_STREAMING_JAR \
 -D mapreduce.job.name="Streaming WordCount" \
 -files count_mapper.py,sum_reducer.py \
 -mapper "python3 count_mapper.py" \
 -reducer "python3 sum_reducer.py" \
 -numReduceTasks $NUM_REDUCERS \
 -input /data/wiki/en_articles_part \
 -output ${OUT_DIR}_tmp > /dev/null &&

# Global sorting as we use only 1 reducer
yarn jar $HADOOP_STREAMING_JAR \
 -D
mapreduce.job.output.key.comparator.class=org.apache.hadoop.mapreduce.l
ib.partition.KeyFieldBasedComparator \
 -D mapreduce.map.output.key.field.separator=. \
 -D mapred.text.key.comparator.options="-k2,2nr" \
 -mapper cat \
 -reducer cat \
 -numReduceTasks 1 \
 -input ${OUT_DIR}_tmp \
 -output ${OUT_DIR}
> /dev/null ) || echo "Error happens"

hdfs dfs -rm -r -skipTrash ${OUT_DIR}_tmp > /dev/null

hdfs dfs -cat ${OUT_DIR}/part-00000 | head
```

4. General Recommendations.

- Use HDFS folders with suffix _tmp for temporary data in HDFS (e.g. my_hdfs_folder_tmp)
- Make sure to remove all temporary data when MapReduce job is complete
- When you have a chain of MapReduce Jobs, please validate status code of the previous jobs before running the next ones.

[2] When you have sep=<tab> and you would like to use 2 columns for a key, you have to change this line:
`-D stream.num.map.output.key.fields=2 \`

- Make sure your STDOUT output complies with Programming Assignment description. Otherwise, tests will fail.
- Even when you work with toy examples, for scalability purposes it is forbidden to load all HDFS output into client node RAM to sort and output data.

5. Submission Rules.

Submission information:

- Short name: **HW2:MapReduce**.
- All your code and output should be located in one folder.
- Script to run your MapReduce Applications should be named **run.sh**:
  - it should read input HDFS-folder from CLI argument $1
  - it should save the HDFS output into folder provided by CLI argument $2 (e.g. `./run.sh /data/input_data hw2_mr_output mr_job_name`)
  - it should clean-up temporary HDFS folders before MapReduce application execution.
  - it should output to terminal (STDOUT) requested amount of lines in expected format[3]
- Save STDOUT output into file named **`hw2_mr.out`**
- Pack your solution into archive **HSU20_<surname>_HW#.zip**, example -- **HSU20_Ivanov_HW2.zip**. If you have Linux or Mac OS and your solution is located in folder "my_solution_folder", then[4]:
  - zip -r hw.zip my_solution_folder/

  On Windows 7/8/10 you can right-click on the folder and choose "send >", in a new window you will see an option "compressed ZIP-folder". Do not forget to rename the archive before submission.
- Make sure that the tree of your project looks the following way:
  - **`| HSU2020_<Surname>_<Name>_HW2.zip`**
  - **`| ---- run.sh`**
  - **`| ---- *.py`**
  - **`| ---- hw2_mr.out`**
- Submission to the Grader:
  - Register at Everest
  - Open submission page: BDT-Grader
  - Choose "Submit Job".
  - Specify "Task ID" as **map_reduce.bigram**
  - Upload your solution to "Task solution"

[3] use "hdfs dfs -cat"
[4] flag -r means - recursively

    - Specify your name for Sender ID
- Send the link to the Grader submission to bigdata_hsu2020@bigdatateam.org with the title “Short name. Surname Name”. Example: “**HW2:MapReduce**. Dral Alexey”.
- Before you send the email, please leave us feedback on the homework: http://rebrand.ly/hsubd2020_feedback_hw02. It will help us to update homework descriptions, estimate your load (how many hours you spend on homeworks) and to choose topics for discussion in upcoming classes.

All questions, comments and suggestions are welcome at telegram chat and course email: bigdata_hsu2020@bigdatateam.org.

Good luck!