CSE427S

HOMEWORK 6

M. Neumann

Due: THU 11 OCT 2018 4PM

Getting Started

Update your SVN repository.

When needed, you will find additional materials for *homework x* in the folder `hwx`. So, for the current assignment the folder is `hw6`.

Hint: You can **check your submission** to the SVN repository by viewing `https://svn.seas.wustl.edu/repositories/<yourwustlkey>/cse427s_fl18` in a web browser.

SUBMISSION INSTRUCTIONS

WRITTEN:

- all written work needs to be submitted electronically in *pdf format*[1] via GRADESCOPE
- provide the following information on *every* page of your `pdf` file:
  - name
  - student ID
  - wustlkey (your `wustlkey` indicates the location (SVN repository) for your code submissions; without this information we will not be able to grade your code!)
- start every problem on a *new page*
- **FOR GROUPS**: make a **group submission** on GRADESCOPE and provide the following information for **all group members** on every page of your `pdf` file: names, student IDs, and location of code submissions (**one** student's `wustlkey`)[2]

CODE:

- code needs to be submitted via SVN repository commit (detailed submission instructions are provided whenever code submissions are required)
- make sure to always use the required *file name(s)* and *submission format(s)*
- **comment your code** to receive maximum credit

[1] Please, **type your solutions** or use **clear hand-writing**. If we cannot read your answer, we cannot give you credit nor will we be able to meet any regrade requests concerning your writing.

[2] It is sufficient to commit code to one SVN repository. If you do not specify the repository for your group's code submission clearly, we will only check the first repository, sorting wustlkeys alphabetically.

Problem 1: Top-10-List of Most Popular Movies (60%)

In this problem we will analyze movie rating data and compute a list of the most popular movies. This is essentially LAB 5! Find more detailed information on how to get the data and implementation specifications in the lab instructions. Use the filenames specified in the lab instructions.

(a) Write a MAPREDUCE program to compute the N most popular movies for the data provided in the `TrainingRatings.txt` file. Use the **sum of the ratings** as measure for popularity! N should be a parameter, that you can provide via the command line. Your program should output the sum of the ratings and the movie title for the top-N-list. Comment your code to receive maximum credit.

Use the following test input and output for a top-3-list for testing and debugging:

Input:

```
8,1148143,2.0
8,1174811,5.0
9,63493,5.0
9,516722,4.0
1,1232582,2.0
5,1631874,4.0
5,721546,4.0
8,2035299,3.0
5,826193,5.0
8,1793777,4.0
3,125713,3.0
```

Output:

```
14 What the #$*! Do We Know!?
13 The Rise and Fall of ECW
9 Class of Nuke 'Em High 2
```

(b) Run your implementation for $N = 10$ on the pseudo-cluster using the `TrainingRatings.txt` provided in the Netflix data. Add the result to your written answer.

Submit the `.java` classes for (a) to the `hw6` folder in your SVN repository. **Add the files to your SVN repo before committing:**

```
$ svn add *.java
$ svn commit -m 'hw6 problem 1(a) submission' .
```

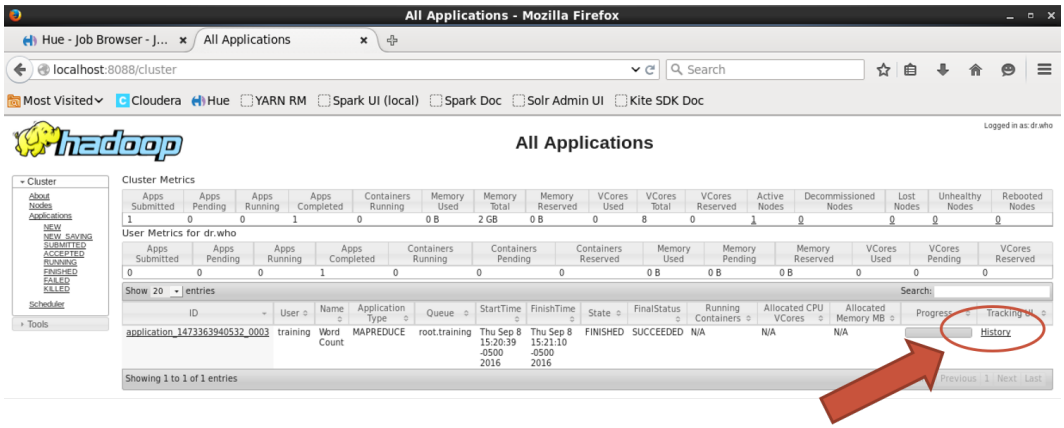
Problem 2: Combiner (40%)

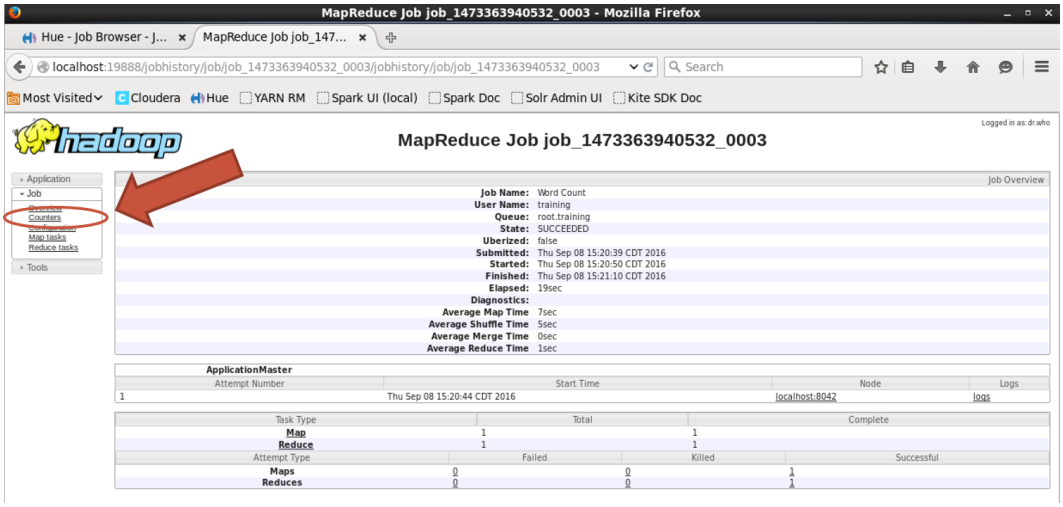
For this problem you will add a Combiner to the first job of your Top-N-List MAPREDUCE program and examine its effect on the job execution.

(a) Add a Combiner to the first job of your Top-N-List implementation from problem 1 (your Combiner should be a "*SumCombiner*"). Provide the respective line(s) of code you added in your driver to your written submission. Does the **result** of your job change compared to the original implementation without Combiner?

(b) Run this job on the `TrainingRatings.txt` data **twice**, once with Combiner (rename your job to "*Aggregation with Combiner*") and once without Combiner (name this job "*Aggregation no Combiner*") and observe the job execution statistics of both jobs.

To do so, open the **YARN Resource Manager Web UI** in a web browser (`http://localhost:8088/`). On this page you will find an overview of completed jobs. Find the ones with names "*Aggregation with Combiner*" and "*Aggregation no Combiner*" and open their **History** (on the very right). Then click on **Counters** (in the panel on the left).





Now, report and compare the total number of counters (last column). Do the following statistics (counters) change? Explain why or why not.

- FILE: `number of bytes read`
- FILE: `number of bytes written`
- HDFS: `number of bytes read`
- HDFS: `number of bytes written`

How many key-value pairs could be combined using the Combiner (HINT:look at those counters: `Combine input records` and `Combine output records`)?

(c) Why can you use the `SumReducer` as Combiner for this job? Give two resons.

(d) Come up with an example Reducer operation (which is **not** the mean/average computation discussed in the lecture) where you cannot use the Reducer as Combiner.

Bonus Problem (5% up to a max. of 100%) - no group work!

Write a review for this homework and store it in the file `hw6_review.txt` provided in your SVN repository (and commit your changes). This file should only include the review, **no other information** such as name, wustlkey, etc. Remember that you are not graded for the content of your review, solely it's completion.

Provide the star rating for `hw6` via this link: `https://wustl.az1.qualtrics.com/jfe/form/SV_aaAc0ILmBleyKzj`.
You can only earn bonus points if you write a meaningful review of **at least 50 words** and provide the corresponding star rating. Bonus points are given to the **owner of the repository only** (no group work!).

Submit your review in the file **`hw6_review.txt`** provided in the `hw6` folder in your SVN repository. **To commit the file run:**

```
$ svn commit -m 'hw6 review submission' .
```

Take 1 minute to provide a star rating for this homework – your star rating should match

Copyrights

The data used in this assignment is taken from Pedro Domingos' class on Data Mining/Machine Learning at University of Washington, 2012.