Lab 4 – The Brightest Stars

CC5212-1 – March 30, 2022

Today, instead of using Hadoop/Java, we will use Pig. This time, for each actor/actress, we will order them (descending) by the number of good movies (according to IMDb ratings) that they star in.

As an example, I will give a Pig script to count the movies co-stars have acted together in (the Pig version of the last lab). This example should help you, but you can also use the slides from the Pig lecture and the Pig documentation: `http://pig.apache.org/docs/r0.14.0/basic.html`. This Pig cheat sheet – `https://www.qubole.com/wp-content/uploads/2014/01/Pig-Cheat-Sheet.pdf` – might also be useful. Instructions for accessing the server are in last weeks lab.

- The data you need are on HDFS in the `/uhadoop/shared/imdb/` folder.
  - The `imdb-stars.tsv` file is the same as last week (about 1GB with 13 million roles); see the schema from last week.
  - There is a new file `imdb-ratings.tsv` that contains the IMDb rating and voting record for over 600,000 movies, TV episodes, etc. The first column gives the distribution of votes (we will not use this). The second column gives the number of votes. The third column gives the mean rating. The other columns give the name of the movie/TV series, the year, the number and the episode name (if any); this part is similar to `imdb-stars.tsv` (but without information on roles).
  - There are also some smaller `*-test.tsv` versions that we will use for creating and debugging our Pig scripts.
- We'll first start by running an example. Grab the project from u-cursos and have a look at the `costar-count.pig` script (in the text editor of your choice). When you understand what it does, change the output directory in the last (STORE) line of the script to your HDFS folder (`/uhadoop2022/`USERNAME`/imdb-costars/`), copy the modified script to the local file system (LFS) of the server in your directory: `/data/2022/uhadoop/`USERNAME. Now we could run the script simply with the command (please don't type this yet) `pig costar-count.pig`. However, it will take some time to run! What happens if your Wifi drops while it's running? ☹. Unix provides a solution called `screen`. It creates a virtual terminal that runs on the server itself, which means you can run something in `screen` on the server, shut down your laptop, go watch a movie, and then come back, fetch the `screen` and see how things are processing. Here's how it works:
  - Type `screen`. A blank terminal starts. Now you can run your Pig script (or any other command!) with `pig costar-count.pig`.
  - If you want to go back to your main terminal, press Ctrl + A, then D (hold Ctrl while pressing A, release both, then press D). Your virtual screen is still running in the background (even if you log out, it will remain). Make sure to take careful note of the number NUM in the message here: `[detached from NUM.pts-3.cluster-01]`. Write it down! This is your screen. There are many like it, but this one is yours. You'll need the number to find the right screen later.
  - If you want to go back to your virtual screen from the main terminal, type `screen -d -r NUM`.
  - If you're done with a virtual screen, while in that virtual screen, press Ctrl + D to kill it (and whatever it was running). Needless to say, it is considered impolite to kill other people's screens.
  - The script will take a while to run, but be sure to retrieve your screen to check on it later!
- The main goal of the lab is to identify the best actors/actresses in IMDb, which we will define as those who acted in the most GOOD MOVIES.
  - We define a GOOD MOVIE as one with ≥ 10001 votes and a score ≥ 7.8.
  - The output will be a count of GOOD MOVIEs for actors and actresses in **two separate files**: one for males, one for females. The output should be in descending order of count.
  - Gender is given as `MALE/FEMALE` in the `gender` column of the `imdb-stars*` inputs.

  - As before, we will only consider entries of type `THEATRICAL_MOVIE`. Again note that in both inputs, `CONCAT(title,‘##’,year,‘##’,num)` is required for a unique movie key.
  - An actor/actress only plays one role in a movie.[1]
  - If an actor/actress does not star in a good movie, **a count of zero should be returned** (rather than omitting the actor/actress from the count).

- From the code project, open the `top-stars.pig` script in the text editor of your choice. This is the one you need to complete. The inputs are given for you so you don't have to figure out the columns; the files loaded are the smaller `*-test.tsv` versions so you don't have to wait too long to see if it's working. The question then is: how should we write the script? There are two options. You can choose whichever you prefer.
  - Write the script like the `costar-count.pig` example. Run it when it's done and it looks like there's no errors. Find errors. Wonder where the errors are in the script. Start outputting the data from each intermediate relation and checking the outputs on HDFS until you find the problematic line. Repeat process until it works (or until the sun comes up, if you decided to follow this option over the full data files).
  - *Or* Build the script in interactive mode line-by-line. Take a small example file (provided by the `-test.tsv` versions of the two inputs). Open the Pig shell (explained later). Write a command to create a new relation `myNextRelation` and hit enter. Call `DESCRIBE myNextRelation` to see what that relation schema looks like. Call `DUMP myNextRelation` to see the data. If all looks okay, copy the command to a text file to create a script. Continue until the output is reached. Later modify the script to read the full file and run it.
    - There are two versions of the Pig shell. You can call `pig` or `pig -x local`. The first will use the MapReduce cluster to run the steps of the Pig script: compiling a .jar, sending it to all machines, allocating the machines, and so forth; if you are playing with a small example, this will slow the process down a lot!! Hence the second option is just to run the Pig tasks locally, which will be a lot faster if you just wish to try on small examples. So use `pig` if you have lots of data and want to use the cluster, otherwise use `pig -x local` if you just want to debug interactively with small examples.
    - If you choose the interactive option, be sure to copy the commands into the `top-stars.pig` script once they are working!
- Once your script is done (whichever option you chose), try testing it on the full data! Change the inputs and outputs of the script to process the full data. Upload the script to your local directory on the master machine. Open a `screen` (like before; you can create a new one if you like) and call `pig top-stars.pig`.
- It's unlikely that you will get a result over the full data before the end of the lab but you can check up on it later. So who is the greatest actor/actress of all time?
- Submit the completed `top-stars.pig` script to u-cursos along with the top 10 actors, top 10 actresses, and 3 random results of an actor/actress you like.
- Questions: How many Map/Reduce phases do we need here?
- Optional: Implement the same task in Hadoop/Java. How does the performance compare?

[1] In reality, multiple roles in the same movie are written on one line like `Role A/Role B`, so they look like one role.