Hadoop Assignment

November 24, 2021

Goals

- *Connect to the cluster and familiarize with the commands.*
- *Run your first Hadoop MapReduce task both locally and on the cluster.*
- *Concatenate MapReduce jobs to perform non-trivial tasks.*

Available Files

- `mapper.py`
- `reducer.py`
- `data/hp_N.txt`
- You can find many other books in txt at `http://www.glozman.com/textpages.html`. Consider concatenating different files to obtain longer books. Be careful with non-ascii characters!

Cluster

The cluster consists of 11 machines, IP addresses between *192.168.0.10* and *192.168.0.19* are the real nodes of the cluster and *192.168.0.20* is the front-end machine that is used to access the cluster resources. Each node of the cluster has the following resources:

- Operating system: CentOS 7
- Storage: 512 GB
- CPU: 6 core Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz
- RAM: 16 GB

Preliminary Activities

1. **Connect to the cluster's front-end.** Follow the instructions below, related to your system, using the following credentials (please replace `<user_lsc_i>` with your group identifier `i`):

   `$ username: user<user_lsc_i> password: <your password>`

   MacOs/Linux

   a) Open a Terminal.

   b) Execute the command: `ssh <user_lsc_i>@130.251.61.97` to connect to the front-end.

Windows

a) Download and install Putty (`https://www.putty.org/`).
b) Open Putty.
c) In the Host Name insert `130.251.61.97` that is the address of the front-end.
d) In the Port field, leave the default value.
e) Connection type, select SSH and click on Open.
f) Another terminal will appear, insert username and password when required (use the credentials provided above).

How to copy files from your local machine to the remote master machine and viceversa

Linux/macOS:

If you want to copy files from your local machine to the remote master machine, use the following command:
`$ scp <local_file> <user_lsc_i>@130.251.61.97: <remote_file>`

If you want to copy files from the remote master machine to your local machine, use the following command:
`$ scp <user_lsc_i>@130.251.61.97: <remote_file> <local_file>`
`<remote_file>` is the relative path of the file in the remote master machine, under your home directory.

Windows:

Download PuttySCP and follow this guide:
`https://it.cornell.edu/managed-servers/transfer-files-using-putty`

Exercise 1: Local WordCount

To run the wordcount example, we need two components: the *mapper* and the *reducer*. The mapper processes one line at time, it splits the line into words (that is, sequences of characters separated by spaces) and for each word emits a key-value pair of `< word, 1>`. The following is an example of output (key, value) pairs:
`<Hello, 1>`
`<World, 1>`

The reducer just sums up the values associated with the same key. Since shuffling is already performed, it linearly sums the values of each key.

To execute the code, execute the following line from the terminal:

```
$ cat data/hp1.txt | python mapper.py | sort -k1,1 | python reducer.py
```

The above command allows *to simulate* a whole MapReduce execution: the shuffling phase is done by the `sort -k1,1` command, see `man sort` for further details.
The expected output should be:

```
" 11
"'Course," 1
"'Oh 1
"'Scuse 1
"). 1
"- 1
"-- 4
"--and 2
"... 5
"...for 1
"0' 1
"A 17
"AAAAAAAAAARGH!" 1
"AAAARGH!" 1
"ALL 1
```

To obtain a smarter output, consider cleaning the input while it is read. To sort in a descending way the output, use the command:

```
$ cat data/hp1.txt | python mapper.py | sort -k1,1 | python reducer.py | sort -k2,2 -nr
```

```
the 3306
to 1827
and 1787
a 1577
of 1235
was 1148
he 1019
Harry 903
in 898
his 893
had 691
-- 687
said 659
at 580
you 578
```

Exercise 2: Wordcount on the Cluster

First of all, we need to move the folder `lab1.zip` to our home on the cluster. On your local machine, execute:
`$ scp lab1.zip <user_lsc_i>@130.251.61.97:~`

To extract the content, use:
`$ unzip lab1.zip`

Then, enter the folder:
`$ cd lab1`

Loading Data on the HDFS.

1. An Hadoop job takes input data from files stored on HDFS and stores the output on HDFS. We therefore create an input directory on the distributed file system:

   ```
   $ hdfs dfs -mkdir -p wordcount/input
   ```
   If a relative path is specified, hdfs considers it relative to the user's home directory on hdfs (`/user/<user-id>/`).

   We can list the content of directories with the -ls option:

   ```
   $ hdfs dfs -ls /user/<user_lsc_i>/wordcount
   ...
   $ hdfs dfs -ls /user/<user_lsc_i>/
   ...
   ```

2. We load our input files into the input directory:

   ```
   $ cd lab1
   $ hdfs dfs -put data/* /user/<user_lsc_i>/wordcount/input
   ...
   $ hdfs dfs -ls /user/<user_lsc_i>/wordcount/input
   ...
   ```

To run the task, execute:

```
$ hadoop jar /usr/local/hadoop/share/hadoop/tools/lib/hadoop-*streaming*.jar -mapper mapper.py
-file ./mapper.py -reducer reducer.py -file ./reducer.py -input /user/<user_lsc_i>/wordcount/input
-output /user/<user_lsc_i>/wordcount/output
```

At the end of the job, Hadoop will print some usage statistics that will be useful to understand how much network traffic the program is creating. Let's take a look to the output (we print only the last ten):

```
$ hdfs dfs -text wordcount/output/* | tail
```

We finally copy the output from HDFS to the local system with:

```
$ hdfs dfs -getmerge wordcount/output/ output.txt
$ cat output.txt
```

Exercise 3: Concatenating Tasks

Exercise A: MapReduce job concatenation. A MapReduce program for transposing the output of the word count program. More precisely, the program should associate each frequency (i.e., the number of times a word appears in the text) with the words with that frequency.

For instance, given the input

```
car 3
the 6
house 3
phone 5
pen 3
glass 3
battery 5
```

the following output has to be generated:

```
3 car pen house glass
5 battery phone
6 the
```

Exercise B: MapReduce job concatenation. A MapReduce program, obtained by modifying the one developed for Exercise A, that counts the number of words with the same frequency.

For instance, given the input

```
car 3
the 6
house 3
phone 5
pen 3
glass 3
battery 5
```

the following output has to be generated:

```
3 4
5 2
6 1
```

Exercise C: MapReduce job concatenation. A MapReduce program for computing maximum frequency and its associated words. Use the output of Exercise A as input directory.

For instance, given the input

```
3 pen car house glass
5 battery phone
6 the
```

the following output has to be generated:

```
6 the
```

Exercise D: MapReduce Joins. A MapReduce program for performing a simple join-like operation. In previous Exercises, you generated the (frequency,terms) and (frequency,NumberOfWords) associations. In this Exercise, join these two datasets to obtain a (NumberOfWords,terms) association.

For instance, given the input

```
3 pen car house glass
5 battery phone
6 the
```

and

```
3 4
5 2
6 1
```

the following output has to be generated:

```
4 car pen house glass
2 battery phone
1 the
```