Lab Assignment 1 **Big Data**
Due Date: 08/01/2019 Prof. P. Boncz
Jeroen Schmidt (122808560) & Dimitris Michailidis (12325929) TA: Dean De Leo
Group 10

Part 1 - Running the Example Script

For the purpose of this tutorial, we created an Elastic MapReduce cluster in Amazon Web Services[1]. To run the Apache Pig script, we invoked the Hue GUI and ran the following script that computes the word distribution of the King James version of the Bible.

```
a = LOAD 'kjv.txt';
b = FOREACH a GENERATE FLATTEN(TOKENIZE((chararray)$0)) AS word;
c = GROUP b BY word;
d = FOREACH c GENERATE COUNT(b) AS ct, group;
e = ORDER d BY ct DESC;
STORE e INTO 'bible_wordcount';
```

The pig script follows the process below:

- Reads the text data from a file called 'kjv.txt' and saves it to `a`.
- Converts `a` into a character array(string) and then splits the string into a bag that contains each word in the text.
- Flattens the bag output of Tokenize into an array and groups it by word.
- Calculates each word's appearance frequency and orders the words by their frequency.
- Saves the output into a folder in the cluster.

The script saves the results in a file "part-r-00000" under the folder bible_wordcount. The file's format is `word _tab_ count`. Below are the first 10 results written into the file:

Word	Frequency
the	72281
and	46774
of	40369
to	16551
that	15704

As we can clearly see, the most frequent words in the text are very common English stop-words.

Part 2 - Extending the Example Script

In this section we modify the script of Part 1 to extend its functionality. Specifically, we add the following 3 extra features:

- Save the dataset in S3 and load it in the Pig script.

[1] https://aws.amazon.com/emr/

- Find the 20 most frequent words with size more than 3.
- Store the results in S3.

We extended the code as follows:

```
a = LOAD 's3n://bd-group10/kjv.txt';
b = FOREACH a GENERATE FLATTEN(TOKENIZE((chararray)$0)) AS word;
b1 = FILTER b BY (int)SIZE(word)>3;
c = GROUP b1 BY word;
d = FOREACH c GENERATE COUNT(b1) AS ct, group;
e = ORDER d BY ct DESC;
f = LIMIT e 20;
STORE f INTO 's3n://bd-group10/bible_wordcount_larger_3_top20';
```

After we flatten the words array, we filter them by size, keeping only those whose size is greater than 3. We then proceed to group them by word as we did before. Before we store the data we order them by count and then limit the selection to the first 20. We finally save the data to S3. The resulting file's top 5 rows are the following:

Word	**Frequency**
that	15704
shall	10872
unto	10063
they	8597
with	7380

We can now see that the words with length of 3 or less are filtered out of our results.

Part 3 - System Description

There are seven layers which are interacted with depending on how you define a layer. See figure 1.

We create an EMR cluster through the AWS interface. This creates a cluster of EC2 instances which have EBS volumes mounted and are connected to an S3 bucket. These EC2 instances also have the necessary configurations and software packages that allow for a hadoop cluster to be created across the EC2 instances. In other words, EMR is a managed hadoop service.

Figure 1 shows the layers and systems that are interacted with in this assignment [2]. The following points explain how the layers are interacted with;

- We run PIG code through the HUE interface.
- This code is then scheduled through oozie (in this case nothing is scheduled and as such the PIG code runs immediately).
- When the PIG code is run, it executes by using the mapreduce layer (as the computer layer) but before this is done it first acquires resources from YARN (the cluster resource manager).

[2]NOTE: HIVE in the diagram is not interacted with in the case where we run PIG, but it is present in the EMR cluster when it is created.

- The PIG (mapreduce) job then runs on the EC2 machine across the cluster.
- The job first reads from our S3 bucket, performs mapreduce using the HDFS file system (defined over the EBS volumes) and then writes the result to the S3 bucket.

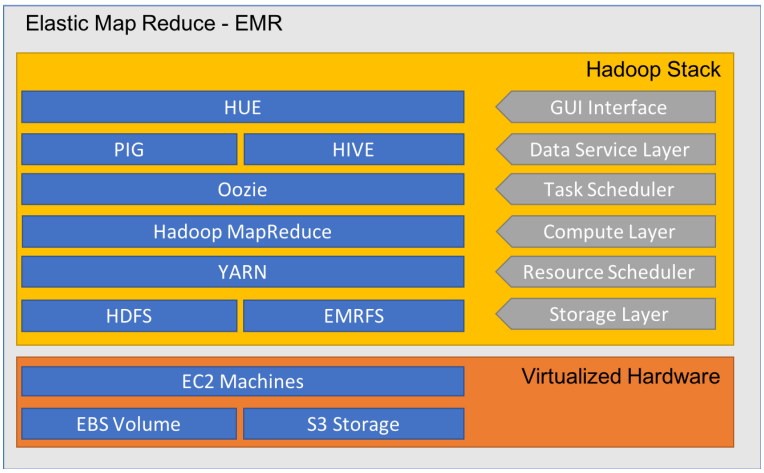

Figure 1: Stack of Interactions

Part 4 - Billing Estimator

This exercise only needs two points to be considered for costing; the EC2 usage and the EMR service usage. The S3, EBS and network usage costs have been ignored because the usage values are negligible enough to be ignored because of the small size of the work load this assignment required.

Resource Used	**Number of Resources**	**$/Hour**	**Hours Used**	**Total $**
EC2: m3.xlarge	3	$0.266	1.15	$0.9177
EMR: m3.xlarge	3	$0.070	1.15	$0.2415
S3	<1Mb	N/A	N/A	0
EBS	N/A	N/A	N/A	0
Network	N/A	N/A	N/A	0
			Total:	$1.159