Homework 1 – HDFS & MapReduce

Requirement:

1. **Install HDFS cluster on multiple machines (at least 3).**

- One of them is Namenode, the other two are Datanodes.
- Configure to have 2 copies (replications)
- Store at least 1GB of data

Reporting list:

- Describe the installation results
- Demo to show the 2 copies were made.
- Demo to show 1GB of data stored.

2. **Run Hadoop applications.**
   General requirements: run the following applications on the systems deployed above.

2.1.Hadoop Word Count

The next program to test is the hadoop word count program. This example reads text files and counts how often words occur. The input is text files and the output is text files, each line of which contains a word and the count of how often it occured, separated by a tab.

Each mapper takes a line as input and breaks it into words. It then emits a key/value pair of the word and 1. Each reducer sums the counts for each word and emits a single key/value with the word and sum. As an optimization, the reducer is also used as a combiner on the map outputs. This reduces the amount of data sent across the network by combining each word into a single record.

Before you can run the example, you'll have to copy some data into the distributed filesystem (HDFS). Here we will create an input directory, and copy in the complete works of Shakespeare and the bible (a standard large corpus for text mining)

The datafile is also avalable at - make sure to gunzip after downloading: [bible+shakes.nopunc.gz]

```
$ hadoop fs -mkdir /user/USERNAME/wordcount

$ hadoop fs -mkdir /user/USERNAME/wordcount/input

$ hadoop fs -put /bluearc/data/schatz/data/textmining/bible+shakes.nopunc
/user/mschatz/wordcount/input
```

To run the example, the command syntax is

```
$ hadoop jar /usr/lib/hadoop/hadoop-examples.jar wordcount \

            /user/USERNAME/wordcount/input \

            /user/USERNAME/wordcount/output
```

After this completes, download the results to your local directory like this:

```
$ hadoop fs -get /user/USERNAME/wordcount/output output
```

Question: What are the top 10 most frequently used words in the corpus?

Hint: Use the unix commands sort and head to scan the output file

2.2. Hadoop Kmer Counting

The next exercise will be to implement a kmer counter using hadoop. Conceptually this is very similar to the wordcount program, but since there are no spaces in the human genome, we will count overlapping kmers instead of discrete words.

The idea is if the genome is:

```
>chr1
ACACACAGT
```

And we are counting 3-mers, your map function will output

```
ACA   1
CAC   1
ACA   1
CAC   1
ACA   1
CAG   1
AGT   1
```

The shuffle function will sort them so the same key comes right after each other

```
ACA   1
ACA   1
ACA   1
CAC   1
CAC   1
CAG   1
AGT   1
```

And your reducer will output:

```
ACA   3
CAC   2
CAG   1
AGT   1
```

You can implement this in Java, using the WordCount program as an example, or you can use Hadoop Streaming to implement it in any language you would like.

The Hadoop Streaming documentation describes how to use it:

https://hadoop.apache.org/docs/r1.2.1/streaming.html

And here is a nice tutorial using Python:

http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/

The genome file is available here: ecoli.fa.gz

Question: What are the top 10 most frequently occurring 9-mers in E coli?