

# Cloud Computing Homework 3: Playing with Hadoop

## Detailed Report

### Task Specifications:

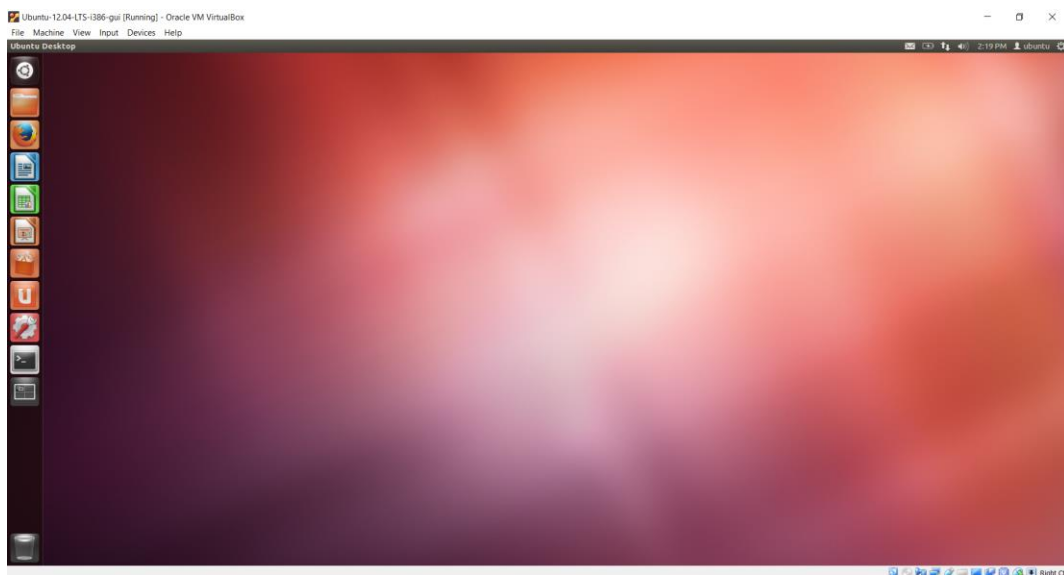
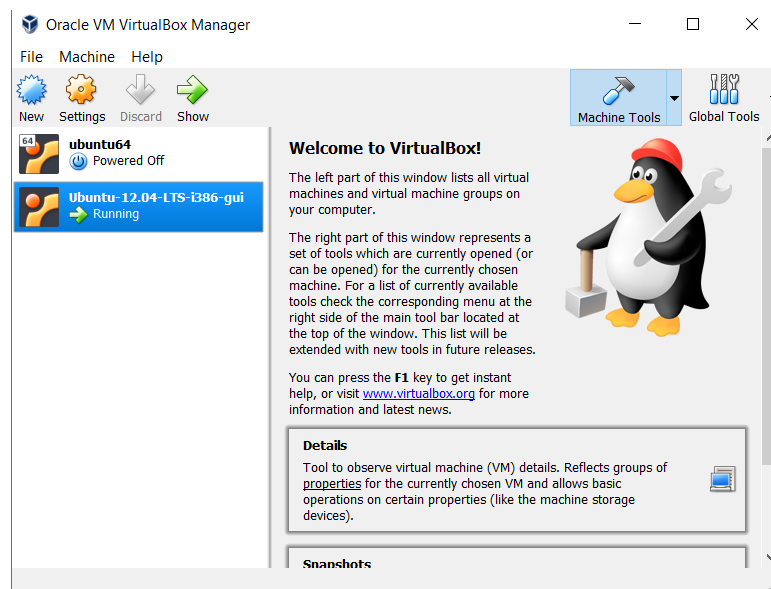
In this homework, I was asked to prepare an environment to work (Step 1), install Apache Hadoop in this environment (Step 2) and go through two coding tutorials (Steps 3 and 4). There was also a bonus task.

### Step 1: Preparing a Linux Environment to Work

**Time Specifications:** 10 min

**Issues:** No Issues Found

**Snapshots:**



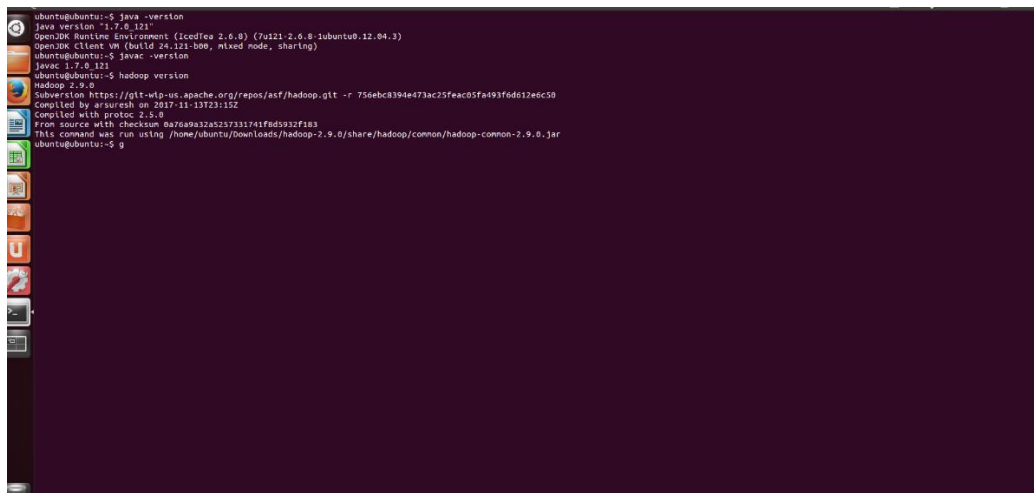
## Step 2: Install Hadoop

**Time Specifications:** 40 Mins

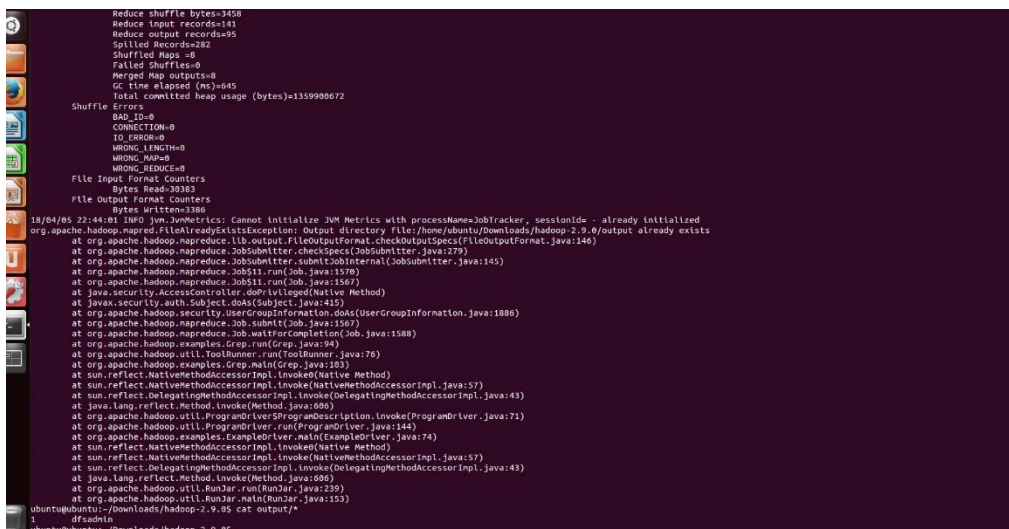
**Issues:** Some Issues Found.

- The instructions given for the Standalone Installation of Hadoop were not helpful.
- Using Hadoop 2.9.0 with Java 1.6 threw version compatibility mismatch error. The Java Version had to be upgraded to 1.7.0\_121.
- Many Java and Hadoop environment variables had to be changed later.
- For eg. Following variables had to be added in ~/.bashrc file.
  - 1) export JAVA\_HOME=/usr/lib/jvm/java-7-openjdk-i386
  - 2) export PATH=\$PATH:\$JAVA\_HOME/bin
  - 3) export HADOOP\_HOME=/home/ubuntu/Downloads/hadoop-2.9.0
  - 4) export PATH=\$PATH:\$HADOOP\_HOME/bin
- The New Java Version had to be set as default.

**Snapshots:**



```
ubuntu@ubuntu:~$ java -version
java version "1.7.0_121"
OpenJDK Runtime Environment (IcedTea 2.6.8) (bun21:2.6.8-1ubuntu0.12.04.3)
OpenJDK Client VM (build 24.121-b06, mixed mode, sharing)
ubuntu@ubuntu:~$ javac -version
javac 1.7.0_121
ubuntu@ubuntu:~$ hadoop version
Hadoop 2.9.0
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r 756ebc8394e473ac25feac05fa93fdd612edc50
Compiled by arsureh on 2017-11-13T23:15Z
Compiled with protoc 2.5.0
From source with checksum 0a76a9a32a5253731741f8d5932f183
This command was run using /home/ubuntu/Downloads/hadoop-2.9.0/share/hadoop/common/hadoop-common-2.9.0.jar
ubuntu@ubuntu:~$ g
```



```
Reduce shuffle bytes=3458
Reduce input records=141
Reduce output records=95
Spilled Records=282
Shuffled Maps=0
Merged Map outputs=0
GC time elapsed (ms)=845
Total committed heap usage (bytes)=1359990672

Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_EXCEPTION=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDCOUNT=0

File Input Format Counters
  Bytes Read=30383
  Bytes Written=1386

18/04/05 22:44:01 INFO jvm.JvmMetrics: Cannot initialize JVM Metrics with processName=JobTracker, sessionId= already initialized
org.apache.hadoop.mapred.lib.OutputCommitter$1: Output directory file:/home/ubuntu/Downloads/hadoop-2.9.0/output already exists
    at org.apache.hadoop.mapreduce.lib.output.FileOutputFormat.checkOutputSpecs(FileOutputFormat.java:146)
    at org.apache.hadoop.mapreduce.JobSubmitter.checkSpecs(JobSubmitter.java:279)
    at org.apache.hadoop.mapreduce.JobSubmitter.submitToInternal(JobSubmitter.java:145)
    at org.apache.hadoop.mapreduce.Job$1.run(Job.java:1570)
    at org.apache.hadoop.mapreduce.Job$1.run(Job.java:1567)
    at java.security.AccessController.doPrivileged(Native Method)
    at javax.security.auth.Subject.doAs(Subject.java:415)
    at org.apache.hadoop.mapreduce.JobSubmitter.doAsUserGroupInformation(JobSubmitter.java:1886)
    at org.apache.hadoop.mapreduce.Job.submit(Job.java:1567)
    at org.apache.hadoop.mapreduce.Job.waitForCompletion(Job.java:1588)
    at org.apache.hadoop.examples.Grep.run(Grep.java:194)
    at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:76)
    at org.apache.hadoop.examples.Grep.main(Grep.java:103)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    at org.apache.hadoop.util.ProgramDriver$ProgramDescription.invoke(ProgramDriver.java:71)
    at org.apache.hadoop.util.ProgramDriver.run(ProgramDriver.java:144)
    at org.apache.hadoop.examples.ExampleDriver.main(ExampleDriver.java:74)
    at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:57)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
    at java.lang.reflect.Method.invoke(Method.java:606)
    at org.apache.hadoop.util.RunJar.run(RunJar.java:230)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:133)
ubuntu@ubuntu:~/Downloads/hadoop-2.9.0$ cat output/*
dfadmin
ubuntu@ubuntu:~/Downloads/hadoop-2.9.0$
```

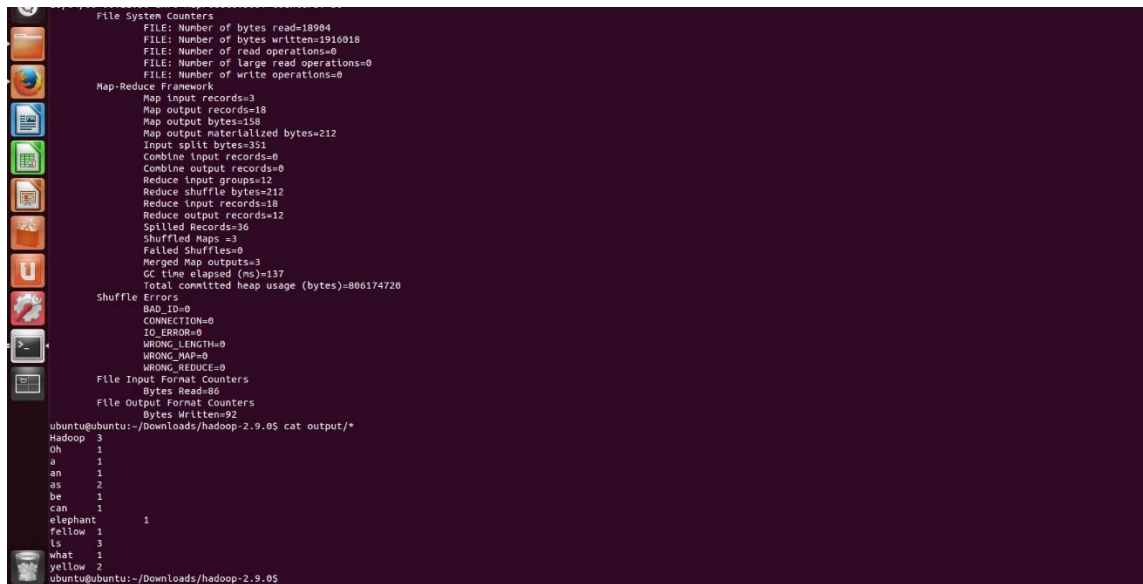
### Step 3: Go through a Simple Hadoop Application

**Time Specifications:** 1 Hour 20 mins

**Issues:** Some Issues Found.

- The commands given in the HW3 Document Tutorial to run a Hadoop Application was using a Cloudera Quickstart VM.
- I did not use the commands as it wasn't necessary because I already had a VM installed and the Hadoop environment up and running. I used the same commands to run a Standalone Hadoop Application as in the previous example and obtained the results.
- I faced a lot of compilation issues of javac. I had to use "javac -cp \$(hadoop classpath) WordCount.java -d build -Xlint" for compilation.

**Snapshots:**



```
File System Counters
  FILE: Number of bytes read=18984
  FILE: Number of bytes written=1916018
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
Map-Reduce Framework
  Map input records=3
  Map output records=18
  Map output bytes=158
  Map output materialized bytes=212
  Input split bytes=351
  Combine input records=0
  Combine output records=0
  Reduce input groups=12
  Reduce shuffle bytes=212
  Reduce input records=18
  Reduce output records=12
  Spilled Records=36
  Shuffled Maps =3
  Failed Shuffles=0
  Merged Map outputs=3
  GC time elapsed (ms)=137
  Total committed heap usage (bytes)=806174720
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=86
File Output Format Counters
  Bytes Written=92
ubuntu@ubuntu:~/Downloads/hadoop-2.9.0$ cat output/*
Hadoop 3
Oh 1
a 1
an 1
as 2
be 1
can 1
elephant 1
fellow 1
ls 3
what 1
yellow 2
ubuntu@ubuntu:~/Downloads/hadoop-2.9.0$
```

### Step 4: Go through a Simple Hadoop Application using Pig

**Time Specifications:** 1 Hour 40 mins

**Issues:** Many Issues Found.

- Used Apache Pig version 0.17.0
- Following Environment Variables had to be added to the ~/.bashrc file:  
export PIG\_HOME="/home/ubuntu/Downloads/pig-0.17.0"  
export PIG\_CONF\_DIR="/home/ubuntu/Downloads/hadoop-2.9.0/etc/hadoop"  
export PIG\_CLASSPATH="\$PIG\_CONF\_DIR"  
export PATH="\$PIG\_HOME/bin:\$PATH"  
export HADOOP\_CONF\_DIR="/home/ubuntu/Downloads/hadoop-2.9.0/etc/hadoop"
- On Execution of Pig Local, I received error that 'JAR does not exist or is not a normal file'. This was solved by deleting two jar files present in the directory and running 'ant jar' in the root directory of Pig.
- For Execution of Pig MapReduce, PIG\_CONF\_DIR and HADOOP\_CONF\_DIR had to be reconfigured to the location of the cluster config directory, i.e. "/hadoop-2.9.0/etc/hadoop/".

## Snapshots:

```
ubuntu@ubuntu:~$ pig -h
Apache Pig version 0.17.0 (r1797386)
compiled Jun 02 2017, 15:41:58

USAGE: Pig [options] [-] : Run interactively in grunt shell.
      Pig [options] -e[execute] cmd [cmd ...] : Run cmd(s).
      Pig [options] [-f[file]] file : Run cmds found in file.

options include:
  -d, -log4jconf - Log4j configuration file, overrides log conf
  -b, -brief - Brief logging (no timestamps)
  -c, -check - Syntax check
  -d, -debug - Debug level, INFO is default
  -e, -execute - Commands to execute (within quotes)
  -f, -file - Path to the script to execute
  -g, -embedded - ScriptEngine classname or keyword for the ScriptEngine
  -h, -help - Display this message. You can specify topic to get help for that
topic.
  -p, -properties - properties is the only topic currently supported: -h properties.
  -i, -version - Display version information
  -l, -logfile - Path to client side log file; default is current working dire
ctory.
  -m, -param_file - Path to the parameter file
  -p, -param - Key value pair of the form param=val
  -r, -dryrun - Produces script with substituted parameters. Script is not exe
cuted.
  -t, -optimizer_off - Turn optimizations off. The following values are suppor
ted:
    ConstantCalculator - Calculate constants at compile time
    SplitFilter - Split filter conditions
    PushdownFilter - Filter as early as possible
    MergeFilter - Merge filter conditions
    PushdownForEachFlatten - Join or explode as late as possible
    LimitOptimizer - Limit as early as possible
    ColumnReplayPrune - Remove unused data
    AddForEach - Add ForEach to remove unneeded columns
    MergeForEach - Merge adjacent ForEach
    GroupbyConstrParallelSetter - Force parallel 1 for "group all" statem
ent
    PartitionFilterOptimizer - Pushdown partition filter conditions to l
oader implementing LoadTestData
    PredicatePushdownOptimizer - Pushdown filter predicates to loader in
plenenting LoadPredicatePushdown
    All - Disable all optimizations
  All optimizations listed here are enabled by default. Optimization value
s are case insensitive.
  -v, -verbose - Print all error messages to screen
  -w, -warning - Turn warning logging on; also turns warning aggregation off
  -x, -executype - Set execution mode: local|mapreduce|tez, default is mapreduc
e.
  -f, -stop-on-failure - Aborts execution on the first failed job; default is
```

## Local Mode Snapshot:

```
ubuntu@ubuntu:~/Downloads/pigtmp$ cat script1-local-results.txt/*
07 new 2.4494897427831788 2 1.1428571428571426
08 pictures 2.04939015319192 3 1.4999999999999998
08 computer 2.4494897427831788 2 1.1428571428571426
08 s 2.545584412271571 3 1.3636363636363635
10 free 2.2657896674010605 4 1.923076923076923
10 to 2.6457513110645903 2 1.125
10 pics 2.794002794004192 3 1.3076923076923075
10 school 2.828427124746189 2 1.1111111111111114
11 pictures 2.04939015319192 3 1.4999999999999998
11 in 2.1572774865200244 3 1.4285714285714284
13 the 3.1309398305840723 6 1.9375
14 music 2.1105794120443453 4 1.6666666666666667
14 city 2.2360679774997902 2 1.6666666666666665
14 university 2.412090756622109 3 1.4000000000000001
15 adult 2.8284271247461903 2 1.1111111111111112
17 chat 2.9104275004359965 3 1.2857142857142854
19 in 2.1572774865200244 3 1.4285714285714284
19 car 2.23606797749979 3 1.3333333333333333
ubuntu@ubuntu:~/Downloads/pigtmp$
```

## MapReduce Mode Snapshots:

```
Success!

Job Stats (time in seconds):
JobID Maps Reduces MaxMapTime MinMapTime AvgMapTime MedianMapTime MaxReduceTime MinReduceTime AvgReduceTime MedianReduceTime Alias Feature Outputs
job_local1173619579_0004 1 1 n/a n/a n/a n/a n/a n/a n/a ordered_uniq_frequency SAMPLER
job_local1787381488_0003 1 1 n/a n/a n/a n/a n/a n/a n/a filtered_uniq_frequency,uniq_frequency1,uniq_frequency2,uniq_frequency3 GROUP_BY
job_local197899259_0002 2 1 n/a n/a n/a n/a n/a n/a n/a hour_frequency1,hour_frequency2 GROUP_BY COMBINER
job_local1586710044_0001 1 1 n/a n/a n/a n/a n/a n/a n/a clean1,clean2,houred,ngramed1,raw DISTINCT
job_local1891554643_0005 1 1 n/a n/a n/a n/a n/a n/a n/a ordered_uniq_frequency ORDER_BY file:///home/ubuntu/Downloads/pigtmp/script1-hadoop-results,

Input(s):
Successfully read 944954 records from: "file:///home/ubuntu/Downloads/pigtmp/excite.log.bz2"

Output(s):
Successfully stored 13530 records in: "file:///home/ubuntu/Downloads/pigtmp/script1-hadoop-results"

Counters:
Total records written : 13530
Total bytes written : 0
Spillable Memory Manager spill count : 0
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local1586710044_0001 -> job_local197899259_0002,
job_local197899259_0002 -> job_local1787381488_0003,
job_local1787381488_0003 -> job_local1173619579_0004,
job_local1173619579_0004 -> job_local1891554643_0005,
```

```

lady diana 2.359033838334352 4 2.052631578947368
video game 2.3750000000000004 4 1.7647058823529413
falcon 2.3884761828476167 4 2.0
scl fl 2.393172105652398 3 1.4999999999999999
clearing house 2.412090756622109 3 1.4000000000000001
hell 2.440118418760475 4 1.6500000000000006
g spot 2.4410577393003097 4 1.7333333333333332
complications 2.4494897427831783 2 1.1428571428571428
uefa cup 2.4494897427831783 2 1.1428571428571428
cemeteries 2.4494897427831783 2 1.1428571428571428
black hole 2.4494897427831783 2 1.1428571428571428
shauna o 2.4494897427831783 2 1.1428571428571428
warez sitez 2.4494897427831783 2 1.1428571428571428
gay cum 2.4494897427831783 2 1.1428571428571428
palsy 2.4494897427831788 2 1.1428571428571426
asian myst 2.4810842175291756 4 1.8125
rhonda 2.52357307257618 3 1.4285714285714282
cock sucking 2.571583846243569 4 1.7058823529411766
xing 2.618614682831909 3 1.4
south east 2.6457513110645983 2 1.125
57 2.6457513110645983 2 1.125
nake 2.6457513110645983 2 1.125
mpeg3 2.6457513110645983 2 1.125
aust 2.6457513110645983 2 1.125
cyberpunk 2.6457513110645983 2 1.125
soaked 2.654953952106303 3 1.3
foto 2.691946385511004 3 1.3571428571428568
shauna 2.771434838459967 4 1.625
bosch 2.794002794004191 3 1.3076923076923077
hottest 2.7950849718747373 3 1.3333333333333333
alimak 2.81858090754796 9 2.782608695652174
red hot 2.82842712474619 2 1.1111111111111112
orgasmic 2.82842712474619 3 1.2222222222222223
jerk off 2.82842712474619 2 1.1111111111111112
bubbles 2.82842712474619 2 1.1111111111111112
plng 2.82842712474619 5 1.8888888888888893
highway patrol 2.82842712474619 2 1.1111111111111112
brothels 2.82842712474619 2 1.1111111111111112
chilton 2.82842712474619 3 1.2222222222222223
live video 2.82842712474619 2 1.1111111111111112
jenni can 3.0 2 1.0999999999999999
cheryl bachnan 3.162277660168379 2 1.0909090909090909
wallpapers 3.2888995827048717 6 1.9333333333333333
app 3.316624790355399 2 1.0833333333333335
bachnan 3.4641016151377557 2 1.0769230769230766
jerk 3.6055512754639896 2 1.0714285714285712
kinberley 3.6055512754639896 2 1.0714285714285712
tu@ubuntu: ~/Downloads/pigtmps

```

## Bonus Task:

### Task Specifications:

In the Bonus Task, I was asked to write a short description on using Hadoop to build an application for analyzing tweet feeds using the Twitter API. The application had to retrieve the tweets for the last hours and the associated attributes for users, places and entities (URLs, hashtags, user mentions) and then process the retrieved information to identify:

- most popular hashtags in the 50 largest cities in the world.
- most popular words in tweets that include a URL for a popular newspaper, magazine or TV show
- most popular video

### Time Specifications: 40 mins

The files obtained from the Twitter APIs contains user, location and actual tweet information and these fields are tab separated. The tweets are generally obtained in JSON format.

1. To get most popular hashtags in the 50 largest cities of the world:
  - i) The tweets obtained for last hours along with the location information can be put in HDFS. Then PIG JSONLoader can be used to load data of the 50 largest cities in the world into PIG.
  - ii) Now the Hashtags need to be extracted from the tweets. Also, we need to have an attribute of the location(city) along with the hashtag from the tweet.
  - iii) Now Map can be used to count the number of hashtags and then probably a local reduce to get local frequency of a hashtag.
  - iv) Lastly, GroupBy and Sort commands (In Descending Order) can be used to display the count of the hashtags with respect to the location.

2. To get most popular words that include a URL for a popular newspaper, magazine or TV show:
  - i) The tweets obtained for last hours can be put in HDFS. Then PIG JSONLoader can be used to load only that data into PIG which includes URL entities.
  - ii) Now a text file with the list of the URLs of a popular newspaper, magazine or TV show can be used to filter in the tweets based on the URLs of that list. In this manner we keep only the tweets of the popular newspaper, magazine or TV show which are in the list.
  - iii) Now all the proper words can be extracted from the filtered tweets.
  - iv) Now, just as in the WordCount Example given in the Homework, we can map reduce to count the number of occurrences of the words extracted from the above tweets and then sort them in descending order to get the most popular words.
3. To get the most popular video:
  - i) The tweets obtained for last hours can be put in HDFS. Then PIG JSONLoader can be used to load only that data into PIG which has a video id/attribute attached to it.
  - ii) Now, the hashtags/user mentions/URLs/ratings describing the videos can be extracted from the tweets to be used for map-reduce processing along with the video ids/attributes in the tweet.
  - iii) The most frequent hashtags/user mentions/URLs/ratings can be found in the set of video ids/attributes formed.
  - iv) The video that has the most hashtags/user mentions/URLs/ratings from the output must be the most popular video.

**Time Taken to Formulate/Consolidate the Report:** 1 Hour

## REFERENCES:

- <https://askubuntu.com/questions/761127/how-do-i-install-openjdk-7-on-ubuntu-16-04-or-higher>
- <https://stackoverflow.com/questions/16263556/installing-java-7-on-ubuntu>
- [https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html#Standalone\\_Operation](https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/SingleCluster.html#Standalone_Operation)
- [https://www.cloudera.com/documentation/other/tutorial/CDH5/topics/ht\\_wordcount1.html](https://www.cloudera.com/documentation/other/tutorial/CDH5/topics/ht_wordcount1.html)
- <https://stackoverflow.com/questions/8396856/classpath-issue-in-hadoop-java>
- <http://pig.apache.org/docs/r0.10.0/start.html#tutorial>
- <https://pig.apache.org/docs/r0.17.0/start.html>
- <https://developer.twitter.com/en/docs>
- <https://acadgild.com/blog/determining-popular-hashtags-in-twitter-using-pig/>
- <http://blog.cloudera.com/blog/2012/09/analyzing-twitter-data-with-hadoop/>
- <http://blog.enablecloud.com/2013/02/find-popular-hash-tags-in-twitter-using.html>
- <https://www.linkedin.com/pulse/100-most-used-english-words-using-hadoop-carlos-iglesias-fern%C3%A1ndez>
- [https://www.packtpub.com/mapt/video/big\\_data\\_and\\_business\\_intelligence/9781787125568/11436/11448/example--most-popular-movie](https://www.packtpub.com/mapt/video/big_data_and_business_intelligence/9781787125568/11436/11448/example--most-popular-movie)