MINI PROJECT REPORT

DATA ANALYSIS IN HADOOP

Team Members:

- ***Rohit Jain (20134133)***
- ***Raman Kr. Banka (20134081)***
- ***Rahul Chaurasia(20134055)***
- ***Samrat Karambir Singh(20134097)***
- ***Pran Nath(20134043)***

B.Tech Third Yr.

Computer Science and Engineering

Motilal Nehru National Institute of Technology,Allahabad

Under the Guidance of:

Miss. Shashwati Banerjea

TABLE OF CONTENTS

Contents

TABLE OF CONTENTS

0. INTRODUCTION

0.1 ABOUT THE PROJECT

In this project we have tried to perform various big data analysis' using hadoop. As hadoop supports processing of structured, semi structured and un structured data efficiently,we found log files are good real time exampes of un structured data, and processing them through hadoop will be the best use case for hadoop in action.
Here We have given sample analysis for 2 of the best possible log files:

- wikimedia project dump
- Store data

0.2 OUR AIMS FROM THIS PROJECT

Our first aim from the project was to learn how Hadoop works, and getting able to write good mapreduce codes. This helped us learn and and now we can use hadoop for analysis of some different problems using a different dataset on our own. We learnt to write our own Mappers and Reducers from scratch. We have also learned to use hadoop streaming along with Python.

0.3 WHAT IS OUR PRIORITY?

Hadoop is not meant for real time stuff on the first place. It is best suitable for batch jobs. the mapreduce framework needs some time to accept and setup the job, which we can't avoid. So our utmost priority while running mapreduce jobs is to get the result right and handle overload on any machine.

1. VOLUNTEER COMPUTING

1.1 WHAT IS VOLUNTEER COMPUTING?

Volunteer computing is an arrangement (a type of distributed system) in which people (volunteers) provide computing resources to projects, which use the resources to do distributed computing and/or storage.

- Volunteers are typically members of the general public who own Internet-connected personal computers. Organizations such as schools and businesses may also volunteer the use of their computers.
- Projects are typically academic (university-based) and do scientific research. But there are exceptions; for example, GIMPS and distributed.net (two major projects) are not academic.

1.2 FEATURES OF VOLUNTEER COMPUTING

Several aspects of the project/volunteer relationship are worth noting:

- Volunteers are effectively anonymous; although they may be required to register and supply email address or other information, they are not linked to a real-world identity.
- Because of their anonymity, volunteers are not accountable to projects. If a volunteer misbehaves in some way (for example, by intentionally returning incorrect computational results) the project cannot prosecute or discipline the volunteer.
- Volunteers must trust projects in several ways:
- The volunteer trusts the project to provide applications that don't damage their computer or invade their privacy.
- The volunteer trusts that the project is truthful about what work is being done by its applications, and how the resulting intellectual property will be used.
- The volunteer trusts the project to follow proper security practices, so that

hackers cannot use the project as a vehicle for malicious activities.

1.3 PROBLEMS WITH VOLUNTEER COMPUTING AND THEIR SOLUTION

Volunteer computing systems must deal with several issues involving volunteered computers their heterogeneity, their churn (the tendency of individual computers to join and leave the network over time), their sporadic availability and the need to not interfere with their performance during regular use. In addition, volunteer computing systems must deal with problems related to correctness:

- Volunteers are unaccountable and essentially anonymous.
- Some volunteer computers (especially those that are overclocked) occasionally malfunction and return incorrect results.
- Some volunteers intentionally return incorrect results or claim excessive credit for results.

One common approach to these problems is replicated computing, in which each job is performed on at least two computers. The results (and the corresponding credit) are accepted only if they agree sufficiently.

1.4 WHAT ARE WE USING?

Hadoop MapReduce framework has commonly been used to perform large-scale data processing, such as social network analysis, data mining as well as machine learning, on cluster computers. However, building a large dedicated cluster for MapReduce is not cost effective if the system is underutilized. To speedup the MapReduce computation with low cost, the computing resources donated from idle desktop/notebook computers in an organization become true potential.

The MapReduce framework we are using can then be implemented into Volunteer Computing environment to allow such data processing tasks to be carried out on the unused computers. Virtualization technology is deployed to resolve the security and heterogeneity problem in Volunteer Computing so that the MapReduce jobs can always run under a unified runtime and isolated environment.

2. BIG DATA

2.1 WHAT IS BIG DATA?

Big data means really a big data (not even terabytes but petabytes of data and much much more), it is a collection of large datasets that cannot be processed using traditional computing techniques. Big data is not merely a data, rather it has become a complete subject, which involves various tools, technqiues and frameworks.

Thus Big Data includes huge volume, high velocity, and extensible variety of data. The data in it can be of three types.

- Structured data : Relational data.
- Semi Structured data : XML data
- Unstructured data : Word, PDF, Text, Media Logs.

2.2 SOURCES OF BIG DATA

IBM's popular definition of big data is what we like the most, The 3V's define big data to its best:

- Volume
- Velocity
- Variety

Examples:-

Big data involves the data produced by different devices and applications. Given below are some of the fields that come under the umbrella of Big Data.

- Black Box Data : It is a component of helicopter, airplanes, and jets, etc. It captures voices of the flight crew, recordings of microphones and earphones, and the performance information of the aircraft.
- Social Media Data : Social media such as Facebook and Twitter hold information and the views posted by millions of people across the globe.
- Stock Exchange Data : The stock exchange data holds information about the 'buy' and 'sell' decisions made on a share of different companies made by the customers.
- Power Grid Data : The power grid data holds information consumed by a particular node with respect to a base station.
- Transport Data : Transport data includes model, capacity, distance and

availability of a vehicle.

- Search Engine Data : Search engines retrieve lots of data from different databases.

2.3 WHY BIG DATA IS A PROBLEM?

Big data is a term for data sets that are so large or complex that traditional data processing applications are inadequate. Challenges include analysis, capture, data curation,search, sharing, storage, transfer, visualization, querying and information privacy.

2.4 HOW HADOOP HANDLES IT

Hadoop allows distributed processing of large datasets across clusters of computers using simple programming models. A Hadoop frame-worked application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage. Thus hadoop can handle big data efficiently.

3. HADOOP

3.1 WHAT IS HADOOP?

The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures.

3.2 MODULES INCLUDED IN THE PROJECT

The project includes these modules:

- Hadoop Common: The common utilities that support the other Hadoop modules.
- Hadoop Distributed File System (HDFS™): A distributed file system that provides high-throughput access to application data.
  The daemons are:
    - Namenode
    - Datanode
    - Secondary Namenode
- Hadoop YARN: A framework for job scheduling and cluster resource management.

  The daemons are
    - Node Manager
    - Resource manager
- Hadoop MapReduce: A YARN-based system for parallel processing of

large data sets.

3.3 OTHER HADOOP-RELATED PROJECTS AT APACHE

- Ambari™: A web-based tool for provisioning, managing, and monitoring Apache Hadoop clusters which includes support for Hadoop HDFS, Hadoop MapReduce, Hive, HCatalog, HBase, ZooKeeper, Oozie, Pig and Sqoop. Ambari also provides a dashboard for viewing cluster health such as heatmaps and ability to view MapReduce, Pig and Hive applications visually alongwith features to diagnose their performance characteristics in a user-friendly manner.
- Avro™: A data serialization system.
- Cassandra™: A scalable multi-master database with no single points of failure.
- Chukwa™: A data collection system for managing large distributed systems.
- HBase™: A scalable, distributed database that supports structured data storage for large tables.
- Hive™: A data warehouse infrastructure that provides data summarization and ad hoc querying.
- Mahout™: A Scalable machine learning and data mining library.
- Pig™: A high-level data-flow language and execution framework for parallel computation.
- Spark™: A fast and general compute engine for Hadoop data. Spark provides a simple and expressive programming model that supports a wide range of applications, including ETL, machine learning, stream processing, and graph computation.
- Tez™: A generalized data-flow programming framework, built on Hadoop YARN, which provides a powerful and flexible engine to execute an

arbitrary DAG of tasks to process data for both batch and interactive use-cases. Tez is being adopted by Hive™, Pig™ and other frameworks in the Hadoop ecosystem, and also by other commercial software (e.g. ETL tools), to replace Hadoop™ MapReduce as the underlying execution engine.

- ZooKeeper™: A high-performance coordination service for distributed applications.

3.4 ABOUT THE VERSION WE ARE USING

We are using Apache Hadoop 2.7.2 in our project. It is a minor release in the 2.x.y release line, building upon the previous stable release 2.7.1.

Complete reference can be found here:
http://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/CommandsManual.html

3.5 WHO USES HADOOP?

A wide variety of companies and organizations use Hadoop for both research and production.

Full list here: http://wiki.apache.org/hadoop/PoweredBy

3.6 HADOOP API'S

- Java Based API
- Hadoop Streaming

4. INSTALLATION OF HADOOP (MULTI NODE)

4.1 CONFIGURING PRE REQUISITES

1. Install Java

```
sudo apt-get update
sudo apt-get install default-jdk
java -version
```

2. Disable IPv6

```
sudo apt-get install vim

sudo vim /etc/sysctl.conf

# disable ipv6
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1

cat /proc/sys/net/ipv6/conf/all/disable_ipv6 ... (should return zero)
```

3. Adding a dedicated Hadoop User

```
sudo addgroup hadoop
sudo adduser --ingroup hadoop hduser
```

4. Install SSH

```
sudo apt-get install ssh
```

5. Give hduser Sudo Permission

```
sudo adduser hduser sudo
```

6. Setup SSH Certificates

```
su hduser
```

```
ssh-keygen -t rsa -P ""
cat $HOME/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys

ssh localhost
```

4.2 ADDING THE JAVA PATH

a. vim ~/.bashrc

```
#HADOOP VARIABLES START
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
export HADOOP_INSTALL=/home/hduser/hadoop
export PATH=$PATH:$HADOOP_INSTALL/bin
export PATH=$PATH:$HADOOP_INSTALL/sbin
export HADOOP_MAPRED_HOME=$HADOOP_INSTALL
export HADOOP_COMMON_HOME=$HADOOP_INSTALL
export HADOOP_HDFS_HOME=$HADOOP_INSTALL
export YARN_HOME=$HADOOP_INSTALL
export
HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_INSTALL/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_INSTALL/lib"
#HADOOP VARIABLES END
```

b. vim /home/hduser/hadoop/etc/hadoop/hadoop-env.sh

```
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64
```

4.3 CONFIGURING TEMPORARY LOCATIONS

```
sudo mkdir -p /home/hduser/tmp
vim /home/hduser/hadoop/etc/hadoop/core-site.xml

<property>
  <name>hadoop.tmp.dir</name>
    <value>/home/hduser/tmp</value>
      <description>A base for other temporary directories.</description>
  </property>

<property>
```

```
    <name>fs.default.name</name>
    <value>hdfs://master:54310</value>
    <description>The name of the default file system.  A URI whose
    scheme and authority determine the FileSystem implementation.  The
    uri's scheme determines the config property (fs.SCHEME.impl) naming
    the FileSystem implementation class.  The uri's authority is used to
    determine the host, port, etc. for a filesystem.</description>
  </property>
```

4.4 CONFIGURING STORAGE AREAS

```
sudo mkdir -p /home/hduser/hadoop_store/hdfs/namenode
//sudo mkdir -p /home/hduser/hadoop_store/hdfs/datanode
sudo chown -R hduser:hadoop /home/hduser/hadoop_store
vim /home/hduser/hadoop/etc/hadoop/hdfs-site.xml

 <property>
    <name>dfs.replication</name>
    <value>2</value>
    <description>Default block replication.
    The actual number of replications can be specified when the file is
created.
    The default is used if replication is not specified in create time.
    </description>
  </property>

  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/home/hduser/hadoop_store/hdfs/namenode</value>
  </property>

  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/home/hduser/hadoop_store/hdfs/datanode</value>
  </property>
```

4.5 CONFIGURING THE MASTER AND SLAVE LINKS

```
sudo vim /etc/hostname

                    #machine name
```

master

sudo service hostname restart
#reopen terminal after this

sudo vim /etc/hosts
#ipv4 machine_name
192.168.15.111 master
192.168.15.102 slave1
vim /home/hadoopuser/hadoop/etc/hadoop/master
master machine name
master

vim /home/hadoopuser/hadoop/etc/hadoop/slaves
#slave machine names
slave1
slave2

4.6 CONFIGURING THE FILE SYSTEM

cp /home/hduser/hadoop/etc/hadoop/mapred-site.xml.template /home/hduser/hadoop/etc/hadoop/mapred-site.xml
vim /home/hduser/hadoop/etc/hadoop/mapred-site.xml

```
<property>
  <name>mapred.job.tracker</name>
  <value>master:54311</value>
  <description>The host and port that the MapReduce job tracker runs
  at.  If "local", then jobs are run in-process as a single map
  and reduce task.
  </description>
</property>
```

vim /home/hadoopuser/hadoop/etc/hadoop/yarn-site.xml

```
<property>
        <name>yarn.nodemanager.aux-services</name>
        <value>mapreduce_shuffle</value>
</property>
<property>
```

```
                <name>yarn.resourcemanager.scheduler.address</name>
                <value>master:8030</value>
</property>
<property>
                <name>yarn.resourcemanager.address</name>
                <value>master:8032</value>
</property>
<property>
                 <name>yarn.resourcemanager.webapp.address</name>
                 <value>master:8088</value>
</property>
<property>
        <name>yarn.resourcemanager.resourcetracker.address</name>
        <value>master:8031</value>
</property>
```

ssh-copy-id -i ~/.ssh/id_rsa.pub slave1

9. Format Hadoop filesystem
   hadoop namenode -format

10. Starting Hadoop
   su hduser
   sudo chown -R hduser:hadoop /home/hduser/hadoop/
   cd /home/hduser/hadoop/sbin
   start-all.sh

5.HADOOP STREAMING

5.1 WHAT IS HADOOP STREAMING

Hadoop streaming is a utility that comes with the Hadoop distribution. The utility allows you to create and run Map/Reduce jobs with any executable or script as the mapper and/or the reducer.

For example:

```
$HADOOP_HOME/bin/hadoop  jar $HADOOP_HOME/hadoop-streaming.jar \
   -input myInputDirs \
   -output myOutputDir \
   -mapper /bin/cat \
   -reducer /bin/wc
```

5.2 WHY HADOOP STREAMING

Hadoop streaming utility allows you to create and run Map/Reduce jobs with any executable or script as the mapper and/or the reducer.

Even though the Hadoop framework is written in Java, programs for Hadoop need not to be coded in Java but can also be developed in other languages like Python or C++ (the latter since version 0.14.1). However, Hadoop's documentation and the most prominent Python example on the Hadoop website could make you think that you must translate your Python code using Jython into a Java jar file. Obviously, this is not very convenient and can even be problematic if you depend on Python features not provided by Jython. Another issue of the Jython approach is the overhead of writing your Python program in such a way that it can interact with Hadoop

5.3 HADOOP STREAMING AND MORE

Hadoop streaming opens door for more new frameworks to be developed with hadoop. With streaming, hadoop can be used with more and more projects and can be incorporated to many more areas like machine learning and real time data analysis.

6. HADOOP JOBS WE HAVE PERFORMED

6.1 WIKIMEDIA PROJECT

6.1.1 INPUT DATA

Link: http://dumps.wikimedia.org/other/pagecounts-raw/

The input data is hourly data of which page on wikipedia or one of the other projects was visited during that time.

Here are a few sample lines from one file:

- fr.b Special:Recherche/Achille_Baraguey_d%5C%27Hilliers 1 624
- fr.b Special:Recherche/Acteurs_et_actrices_N 1 739
- fr.b Special:Recherche/Agrippa_d/%27Aubign%C3%A9 1 743
- fr.b Special:Recherche/All_Mixed_Up 1 730
- fr.b Special:Recherche/Andr%C3%A9_Gazut.html 1 737

In the above, the first column "fr.b" is the project name. The following abbreviations are used:

- wikibooks: ".b"
- wiktionary: ".d"
- wikimedia: ".m"
- wikipedia mobile: ".mw"
- wikinews: ".n"
- wikiquote: ".q"
- wikisource: ".s"
- wikiversity: ".v"
- mediawiki: ".w"

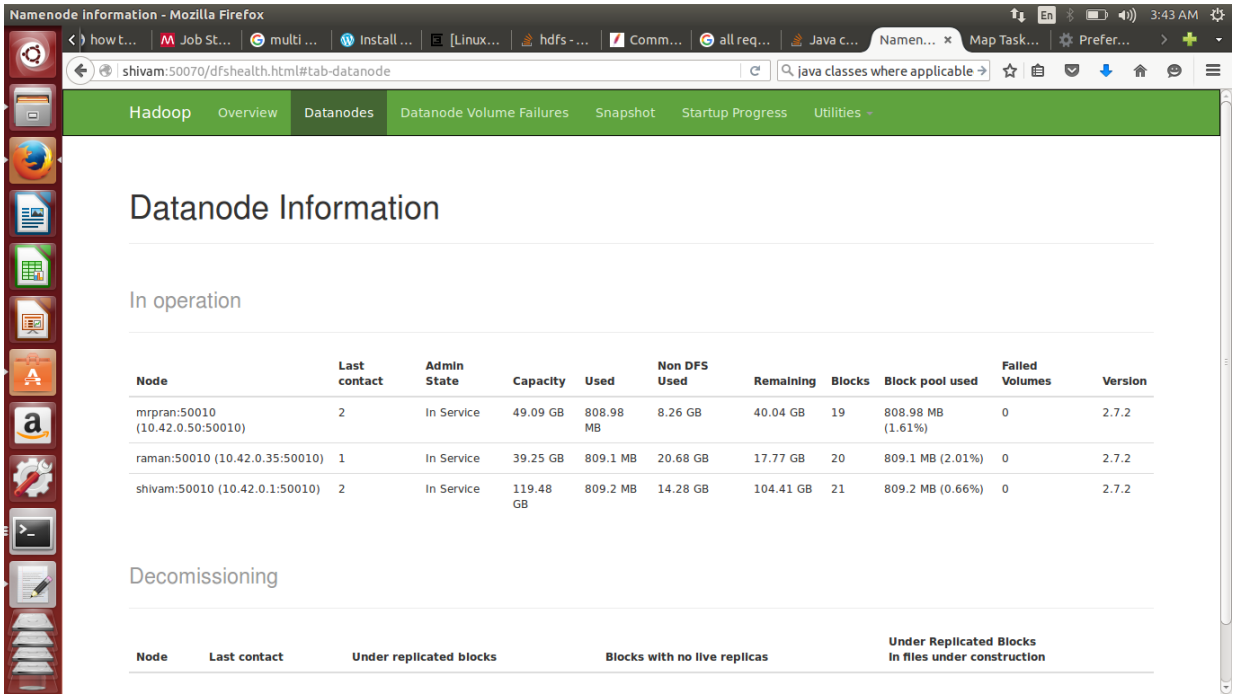
Projects without a period and a following character are wikipedia projects.

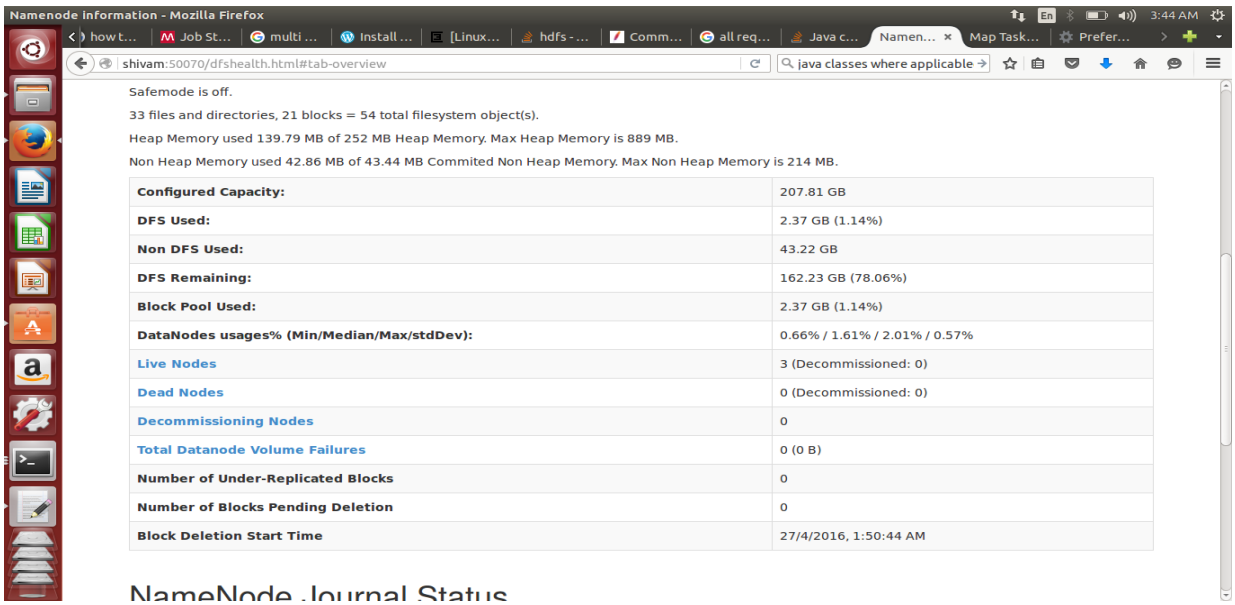
The second column is the title of the page retrieved, the third column is the number of requests, and the fourth column is the size of the content returned.

These are hourly statistics, so in the line

en Main_Page 242332 4737756101

we see that the main page of the English language Wikipedia was requested over 240 thousand times during the specific hour.





6.1.2 QUESTION

We need to find out the Most visited page for each language

6.1.3 MAPPER AND REDUCER

Mapper:

```
#!/usr/bin/env python

#The sheabng line above is necesarry. It lets the hadoop streaming khow about which scripting language to use.

#import the system library
import sys

# Read each line from STDIN
for line in sys.stdin:


  #strip whitespaces from the begining and ending of line and split lines on whitespaces
  data = line.strip().split()

  #if 4 fields are not there, then continue
  if len(data) != 4 :
        continue

  #assign respective values from the data
  language,page,views,size = data

  #print the values to stdout
  print '{}\t{}\t{}'.format(language,page,views)
```

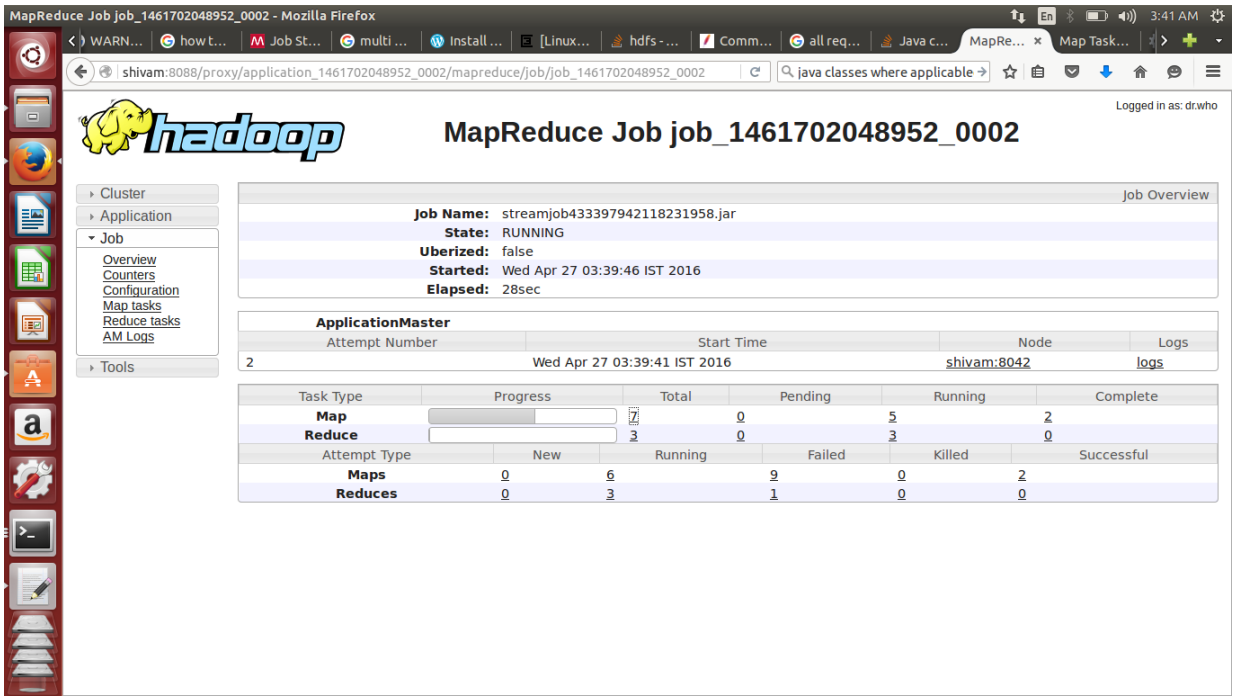
Reducer:

```
#!/usr/bin/python
import sys

oldLanguage=None
oldPage=None
oldViews=None

for line in sys.stdin:
	data = line.strip().split("\t")
	language,page,views = data

	if(len(data)!=3):
		continue

	if (not oldLanguage):
		oldLanguage=language
		oldViews=views
		oldPage=page

	if(oldLanguage!=language):
		print("{}\t{}\t{}".format(oldLanguage,oldPage,oldViews))
		oldLanguage=language
		oldViews=views
		oldPage=page
	if(views > oldViews):
		oldViews=views
		oldPage=page

if(oldLanguage != None):
	print("{}\t{}\t{}".format(oldLanguage,oldPage,oldViews))
```

6.1.4 ANALYSIS

Job Summary:

jobId=job_1459504597536_0007,
submitTime=1459519790857,
launchTime=1459519798406,
firstMapTaskLaunchTime=1459519800959,
firstReduceTaskLaunchTime=1459519827568,
finishTime=1459520579857,
resourcesPerMap=1024,
resourcesPerReduce=1024,
numMaps=7,
numReduces=2,
user=hduser,
queue=default,
status=SUCCEEDED,
mapSlotSeconds=409,
reduceSlotSeconds=1501,
jobName=streamjob2963986293103182895.jar



Mapper Analysis:

1.

{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000000","taskType":"MAP","startTime": ,"splitLocations":"192.168.15.102,192.168.15.114"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000000","taskType":"MAP","finishTime":1459519830262,"status":"SUCCEEDED"}}}

Total time : 31.732s

2.

{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000001","taskType":"MAP","startTime":1459519798520,"splitLocations":"192.168.15.114,192.168.15.102"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000001","taskType":"MAP","finishTime":1459519830252,"status":"SUCCEEDED"}

Total time : 31.732s

3.

{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000002","taskType":"MAP","startTime":1459519798522,"splitLocations":"rohit,raman"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000002","taskType":"MAP","finishTime":1459519876232,"status":"SUCCEEDED"}}}

Total time: 1m17.71s / 77.71s

4.
{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000003","taskType":"MAP","startTime":1459519798523,"splitLocations":"shivam,raman"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000003","taskType":"MAP","finishTime":1459519865340,"status":"SUCCEEDED"}}}

Total Time : 01m 6.81699s or 66.82s

5.
{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000004","taskType":"MAP","startTime":1459519798532,"splitLocations":"rohit,raman"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000004","taskType":"MAP","finishTime":1459519855235,"status":"SUCCEEDED"}}}

Total time: 56.7029s
6.

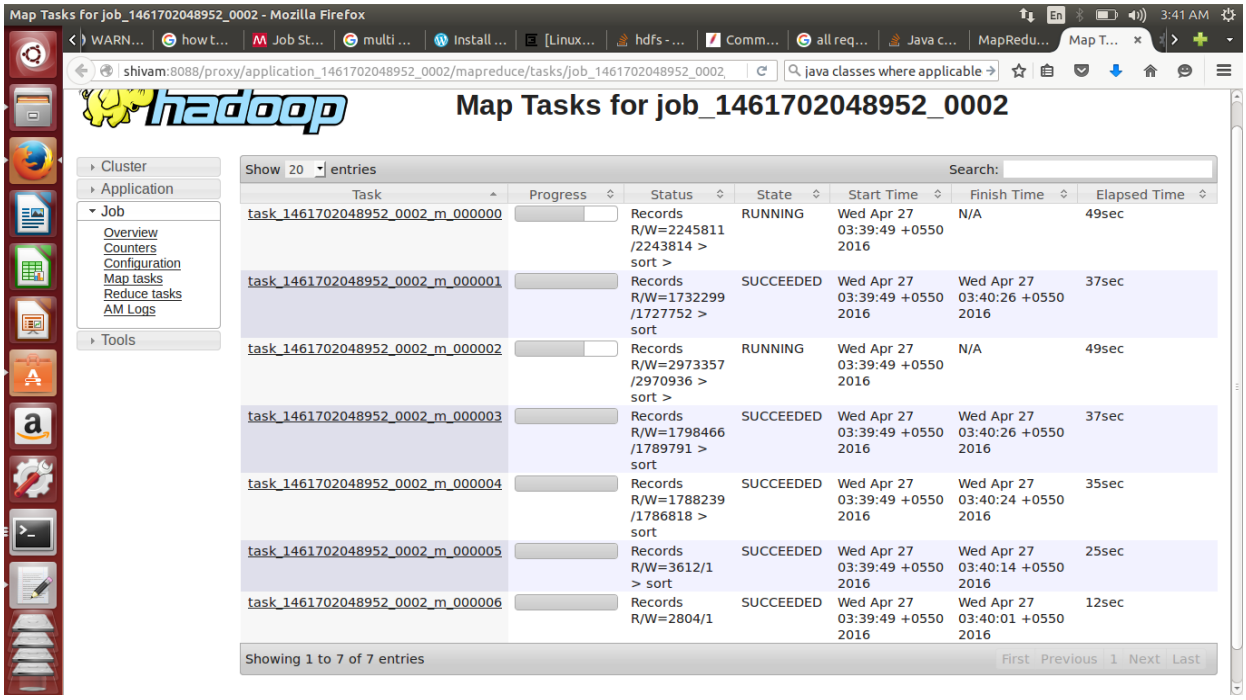
{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000005","taskType":"MAP","startTime":1459519798533,"splitLocations":"rohit,raman"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000005","taskType":"MAP","finishTime":1459519870272,"status":"SUCCEEDED"}}}

Total Time : 01m 11.739s or 71.74s

7.
{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_m_000006","taskType":"MAP","startTime":1459519798533,"splitLocations":"rohit,raman"}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_m_000006","taskType":"MAP","finishTime":1459519823959,"status":"SUCCEEDED"}}}
Total Time: 25.426s



Reducer Analysis:
1.

{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_r_000000","taskType":"REDUCE","startTime":1459519798534,"splitLocations":""}}}

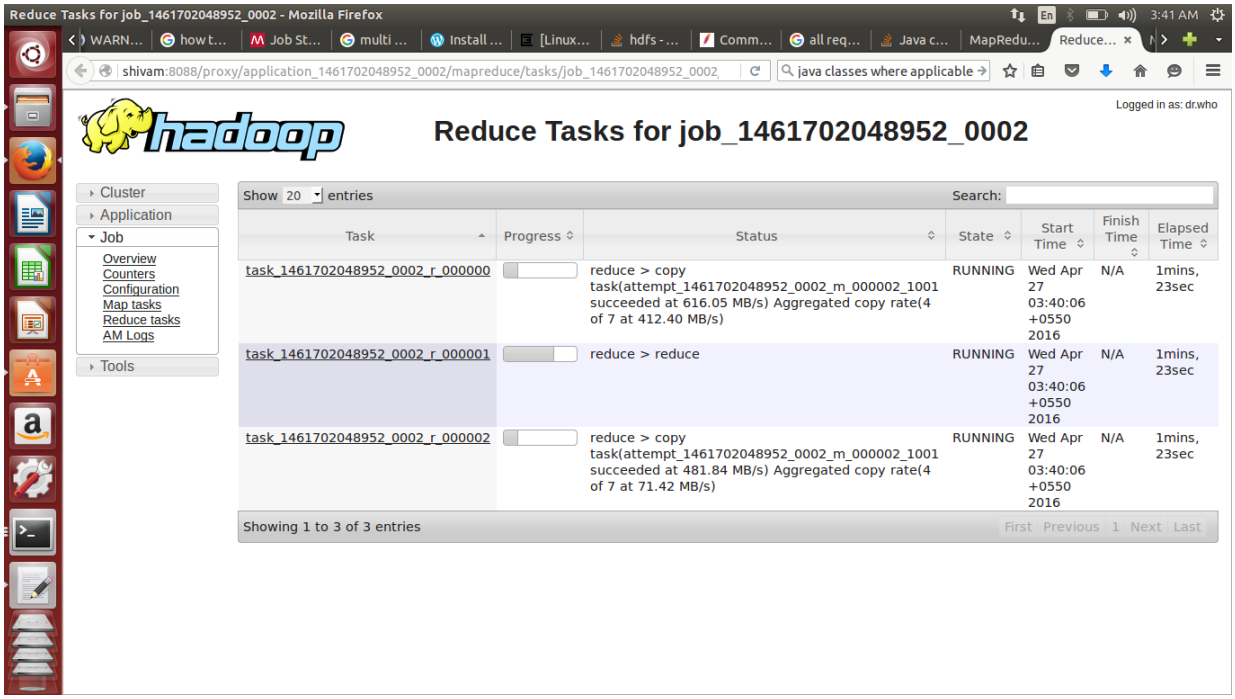
{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_r_000000","taskType":"REDUCE","finishTime":1459520579630,"status":"SUCCEEDED"}}}

Total Time: 13m 1.096s or 781.0s

2.
{"type":"TASK_STARTED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskStarted":{"taskid":"task_1459504597536_0007_r_000001","taskType":"REDUCE","startTime":1459519798534,"splitLocations":""}}}

{"type":"TASK_FINISHED","event":{"org.apache.hadoop.mapreduce.jobhistory.TaskFinished":{"taskid":"task_1459504597536_0007_r_000001","taskType":"REDUCE","finishTime":1459520203107,"status":"SUCCEEDED"}}}

Total time: 06m 44.573s or 404.57s



6.1.4 OUTPUT

The respective output for hourly most visited page was obtained.
The result is too large to be shown here:
Can be found here: www.github.com/ramanbanka/hadoop

6.2 STORE DATA

6.2.1 INPUT DATA

Input data is a store data dump of various stores in usa.
Each line of the input file is of form:
date time store_name product_category price method_of_payment
(delimited by tabs)

6.2.2 QUESTION

1. find out the total sales for each store
2. find out the total sales for each product_category
3. Find the monetary value for the highest individual sale for each separate store.
4. Find the total sales value across all the stores, and the total number of sales. Assume there is only one reducer.

6.2.3 MAPPER AND REDUCER

All files on : http://www.github.com/ramanbanka/hadoop

6.2.4 OUTPUT

All files on : http://www.github.com/ramanbanka/hadoop

7. DESIGN PATTERNS ANALYZED

7.1 FILTERING PATTERNS

- Sampling Data
- Top N of a data set

7.2 SUMMERIZATION PATTERNS

- Counting
- Min/Max
- Statistics
- Index

7.3 STRUCTURAL PATTERNS

- Combining Data Sets

8. HANDLING THE FILESYSTEM

8.1 SCRIPT WE HAVE WRITTEN

```
E_File='/home/hduser/hadoop/conf/dfs.exclude'
tmp='/home/hduser/hadoop/conf/tmp'
if [ "$#" -eq 2 ]
 then
  if [ "$1" == "exclude" ]
    then
      echo $2 >> $E_File
  elif [ "$1" == "include" ]
     then
       sed "/$2/d" $E_File > $tmp
       sed "/^$/d" $tmp > $E_File
   fi
else
  echo "wrong format. Usage :<include/exclude> <datanode_name>"
fi

hadoop dfsadmin –refreshNodes
hadoop mradmin -refreshNodes
```

8.2 DECOMMISIONING NODES

- Shut down the NameNode.
- Set dfs.hosts.exclude to point to an empty exclude file.
- Restart NameNode.
- In the dfs exclude file, specify the nodes using the full hostname or IP or IP:port format.
- Do the same in mapred.exclude
- execute bin/hadoop dfsadmin -refreshNodes. This forces the NameNode

to reread the exclude file and start the decommissioning process.

- execute bin/hadoop mradmin -refreshNodes
- Monitor the NameNode and JobTracker web UI and confirm the decommission process is in progress. It can take a few seconds to update. Messages like "Decommission complete for node XXXX.XXXX.X.XX:XXXXX" will appear in the NameNode log files when it finishes decommissioning, at which point we can remove the nodes from the cluster.

8.3 HANDLING FAILURES

Namenode handles failures of datanodes by heartbeat protocol. NameNode periodically receives a Heartbeat and a Blockreport from each of the DataNodes in the cluster. Receipt of a Heartbeat implies that the DataNode is functioning properly. A Blockreport contains a list of all blocks on a DataNode. When NameNode notices that it has not recieved a hearbeat message from a data node after a certain amount of time, the data node is marked as dead. Since blocks will be under replicated the system begins replicating the blocks that were stored on the dead datanode. The NameNode Orchestrates the replication of data blocks from one datanode to another. The replication data transfer happens directly between datanodes and the data never passes through the namenode.

However, if namenode fails (and there is no backup) then all data is permanently lost. So Namenode is generally built on reliable machines whereas datanotes are usually commodity hardware.

9. REFERENCES

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

https://en.wikipedia.org/wiki/Volunteer_computing

https://boinc.berkeley.edu/trac/wiki/VolunteerComputing

http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=6841858&abstractAccess=no&userType=inst

https://en.wikipedia.org/wiki/Big_data

http://blog.sqlauthority.com/2013/10/02/big-data-what-is-big-data-3-vs-of-big-data-volume-velocity-and-variety-day-2-of-21/

http://hadoop.apache.org/

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