CS252 Project Report and Documentation

Ayush Sekhari
Roll 12185
ayusek@iitk.ac.in

Kaviti Sai Saurab
Roll 12349
kssaurab@iitk.ac.in

Kundan Krishna
Roll 12374
kkrishna@iitk.ac.in

Satvik Gupta
Roll 12633
satvikg@iitk.ac.in

Saurav Shekhar
Roll 12640
sshekh@iitk.ac.in

Sharbatanu Chatterjee
Roll 12658
sharbatc@iitk.ac.in

November 27, 2014

Abstract

This project is part of the course CS252 : Computing Laboratory II. We have worked on integrating R with Hadoop and provide a platform for distributed R-programming. The final result will be a fully functional hadoop cluster capable of running R programs over it. We will also provide an image installed with softwares required to manage and re-configure the system over bladeserver.

1 Problem Statement

The problem we looked at involves the setting up of a R-Hadoop system for the institute. This allows the distributed processing of R-code on a number of clusters. After setting up a cluster containing 4 nodes, on Hadoop, we expect an input file of a piece of code written in R and the system would processs the data in parallel and calculate the result. There will be a wrapper around the entire system such that a user can simply run a single command and get all the possible packages and background work. This would allow the user to run a single script with the input folder, output folder and the name of the R program as arguments and run the entire program using hadoop at once.

2 Theory and Methodology

Hadoop, developed by Apache Software Foundation[2], is an open-source software framework for distributed storage and distributed processing of data on clusters of commodity hardware. The Hadoop Distributed File System (HDFS) divides the input file into large blocks and distributes the blocks among the nodes in a cluster that has been set up. The Hadoop Map/Reduce code shifts code, notably Jar files, into the nodes that have the required data, and the nodes then process the data in parallel.

R is a free software programming language and software environment for statistical computing and graphics. The R language is widely used among statisticians and data miners for developing statistical software. The rising popularity and usage of R has led us to believe that setting up R on Hadoop is essential and important for the department and the institute.

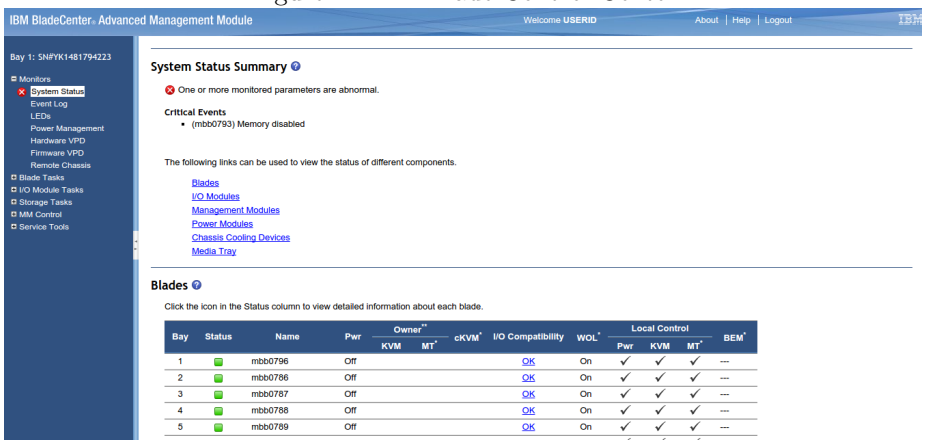
R-on-Hadoop - Integrating R on Hadoop using RHadoop[1] would allow us to set up the analytic system that would help us in solving our problem statement.

IBM Blade Servers

A blade server is a stripped down server computer with a modular design optimized to minimize the use of physical space and energy. Whereas a standard rack-mount server can function with (at least) a power cord and network cable, blade servers have many components removed to save space, minimize power consumption and other considerations, while still having all the functional components to be considered a computer.[clarification needed][1] A blade enclosure, which can hold multiple blade servers, provides services such as power, cooling, networking, various interconnects and management. Together, blades and the blade enclosure form a blade system. Different blade providers have differing principles regarding what to include in the blade itself, and in the blade system altogether.¡br¿

In a standard server-rack configuration, one rack unit or 1U—19 inches (480 mm) wide and 1.75 inches (44 mm) tall—defines the minimum possible size of any equipment. The principal benefit and justification of blade computing relates to lifting this restriction so as to reduce size requirements.

Figure 1: IBM Blade Control Center



Using Hadoop Distributed File System(HDFS)

The HDFS file system stores files across different nodes and keeps multiple copies as a backup, incase a node fails. Hadoop file system interface is just like linux file systems and it gives a host of linux-like functions to interact with the

directories and files. [4]
For example, the command `hadoop dfs -ls /user/hduser` will show you the files in the directory `/user/hduser/`, and their permissions.

```
hduser@cs252-Not-Specified:~$ hadoop dfs -ls /user/hduser
DEPRECATED: Use of this script to execute hdfs command is deprecated.
Instead use the hdfs command for it.

14/11/27 22:09:03 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your pl
Found 8 items
drwxrwxrwx   - hduser supergroup          0 2014-11-26 11:16 /user/hduser/InputData4
drwxrwxrwx   - hduser supergroup          0 2014-11-26 11:40 /user/hduser/OutputData4
drwxrwxrwx   - hduser supergroup          0 2014-11-26 11:17 /user/hduser/teraInput
drwxrwxrwx   - hduser supergroup          0 2014-11-26 12:30 /user/hduser/teraOutput
drwxrwxrwx   - hduser supergroup          0 2014-11-26 12:08 /user/hduser/testData
drwxr-xr-x   - root   supergroup          0 2014-11-26 12:55 /user/hduser/testOutput
drwxrwxrwx   - hduser supergroup          0 2014-11-26 11:56 /user/hduser/wordInput
drwxrwxrwx   - hduser supergroup          0 2014-11-26 11:57 /user/hduser/wordOutput
hduser@cs252-Not-Specified:~$
```

Figure 2: Output of ls command in hdfs

2.1 Methodology

- Installing Ubuntu on Blade servers
  - To set up the blade servers, proceed to the site corresponding to 172.27.18.108, which is the IBM BladeCenter Advanced Management Module of your blade servers.
  - Login (At the time of preparation of this documentation, login:'USERID' password:'PASSW0RD' without quotes, it is the numeral zero and not O in PASSW0RD)
  - Click continue
  - On the lefthand pane goto Blade Tasks→Remote Control
  - Under Remote Control Status, set KVM Owner and Media Tray Owner to the Blade Server in which you want to install Ubunutu 12.04 (Later versions are at the moment not compatible as seen by us after a lot of failed efforts to install Ubuntu 14.04. Apparently, the disk system of the IBM Blade Servers can not be formatted or partitioned by the Ubuntu 14.04 installer). We have included an image in our virtual machine.
  - Start Remote Control
  - select Remote Drive
  - Goto select image option of Available Resources → select the image of ubuntu from your system →select Mount All→close the window
  - Click on PowerControl and select Restart
  - Wait for some time and you will see a blue box saying LS20, keep pressing F12 when you see this
  - Wait and a blue screen opens up to select a device→select cd drive and Enter

  - Ubuntu boots up from the image
  - Proceed with the Ubuntu installation as usual

- Install Hadoop on the blade servers.
  - Installing of Hadoop was done on a single node cluster system initially. three
  - Hadoop framework is written in Java and thus the Java OpenJDK Runtime Environment is installed in the system.
  - A dedicated user is created in the Ubuntu system using `sudo addgroup hadoop` and then `sudo adduser --ingroup hadoop hduser` This user will be the one too use our system to run R progrmas.
  - ssh keys are generated to allow hassle free communication between nodes i.e. remote machines plus our local machine. This would allow hadoop users to use ssh without requiring any password. ssh-keys are generated by using the following command: `ssh-keygen -t rsa -P` which creates a public key to be shared and a private identification saved locally.
  - The zipped package for installing Hadoop is downloaded from `http://mirrors.sonic.net/apache/hadoop/common/hadoop-2.4.1/hadoop-2.4.1.tar.gz` and this is unzipped using the command `tar xvzf hadoop-2.4.1.tar.gz`
  - To set up hadoop, we need to simply move the hadoop installation to the /usr/local/hadoop directory and set the permission for the installation to be owned by the user hduser or anyone.
  - Hence, configure the following files to set up Hadoop system completely:
    1. `~/.bashrc`
    2. `/usr/local/hadoop/etc/hadoop/hadoop-env.sh`
    3. `/usr/local/hadoop/etc/hadoop/core-site.xml`
    4. `/usr/local/hadoop/etc/hadoop/mapred-site.xml.template`
    5. `/usr/local/hadoop/etc/hadoop/hdfs-site.xml`
  - Now the correct environmental variables are set un the ~/.bashrc file. *Note that this would also have to be changed later due to the modifications of using RHadoop* Also be careful that the paths of each of the environmental variables are set correctly. We also have to make sure when we are running the R script, we are running the Sys.setenv() commands also, before our R code. This was done by appending the commands to the .bashrc files, after accessing the R program. This prevents the addition of the commands at every instance of an R shell being called.
  -

- Set up one master node and several slave nodes.
  - After setting up the Hadoop

- Set up stable IP's on the blade server to allow ssh-access to all in institute.

- Ensure working of Mad/Reduce, run a sample code of, say π value approximation calculation.
- Install RHadoop and set up RHadoop on the system.
- Write and run R-code.

2.1.1 From a single node cluster to a multinode cluster

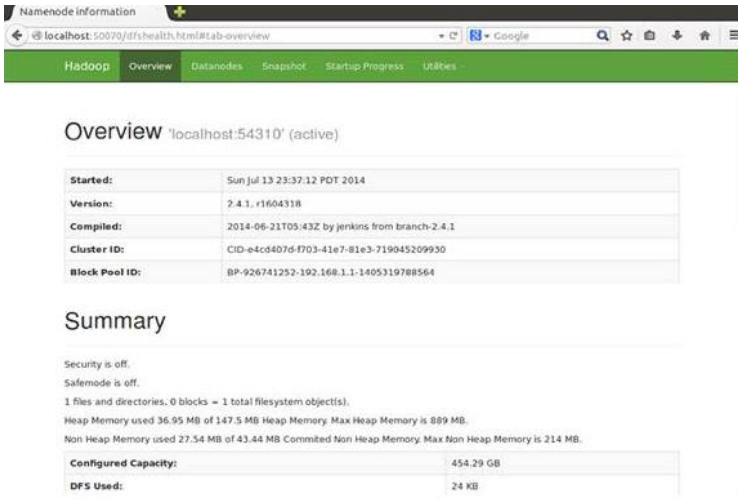
- Open `/etc/hosts` file of all the servers and add the ip followed by a name given to the server for reference. For example, entry added for master is `172.27.18.24 master`.
- We have to add the ssh key of master to `/.ssh/authorized_keys` file of all servers so that we can ssh between different nodes without password.
- Enter all the names assigned to different servers in the `etc/hadoop/slaves` file
- Change the core-site.xml, mapred-site.xml, hdfs-site.xml appropiately. Format the namenode and start the multinode cluster.

2.2 Issues during Hadoop installation

- The first issue was that some of the nodes of Multi-cluster goes down after some time because of some network problem. In order to resolve that just start any website that refreshes itself periodically like espncricinfo.com on all the servers.
- Other issue was even though the servers are connected to the internet they were not connected internally. We resolved it by continuously pinging slaves from master.
- One another issue was that after installing and running hadoop on a node, it gave a 'connection refused' error when connecting to the master in the multi-node cluster and command `jps` in various nodes did not show the datanode in process. To curb this, we had to create a new namenode, datanode on the node to be connected and then format the namenode again.[3]

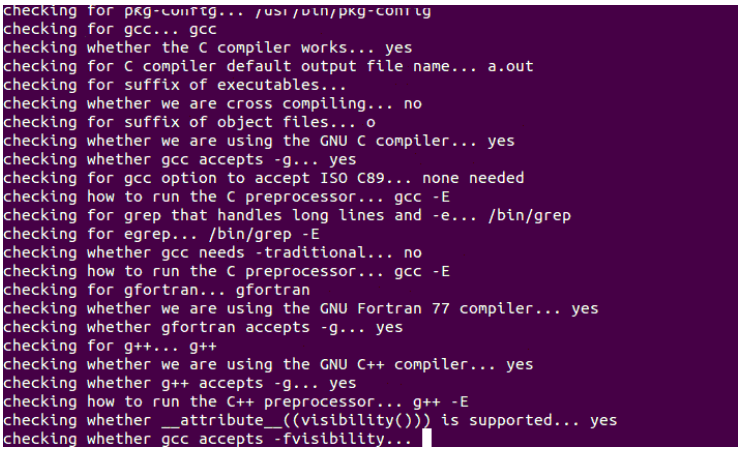
2.3 Checking the correctness of the Cluster

- The command `hadoop dfsadmin -report` shows various datanodes that are active at that time.
- We can also see the active datanodes using WebBrowser. Just open the WebBrowser and type "localhost:50070"



3 Setting up R

- Though you can install R by running `sudo apt-get install r-base`, you would get an older version of R on Ubuntu 12.04 repositories. Hence we decided to go for manually installing the packages.
- We installed R version 3.1. However, before installing the package manually, a lot of dependencies are required to be installed. So we did the following:
  - Install older version of R by running `sudo apt-get install r-base`.
  - Remove R through `sudo apt-get remove --purge r-base-core`. (This removes R but keeps all the dependencies).
  - Download `r-base_3.1.0.orig.tar.gz` (A copy has been included in the Virtual Machine Image that we have submitted).
  - Run `tar -xzvf r-base_3.1.0.orig.tar.gz`.A new folder named `R-3.1.0` will be created. Change directory into that.
  - Run `sudo ./configure`. This checks your system for required prerequisite packages.(Shown in the figure below)



  - Then run `sudo make` followed by `sudo make install`.

- Now you can open up a root bash shell and run the command R to start it.

Setting uo R on Hadoop

- to integrate R on Hadoop set up the requisite packages with downloads sourced from proper servers.
- run the following program script to automate the downloading of the packages (insert sekhari's code)
- some of the packages have dependencies on others so the ordering of the downloads should not be tampered with.
- when prompted to select a server select either the USA Tenessee server (93) or the Vietnam server (83) which will be preferred.
- hence run the commands to install the packages rhdfs, rmr2 and plyrmr in this order. ' hence run the following code to test the working of R by setting the environment variables.
- there are two ways to add the variables in a sureshot way. Either add them by putting export .... in the default .bashrc folder of each script or run the command sys.setenv.... in the R script
- hdfs.init() checks the correct status of set up before every script run.

4 Setting up RHadoop on server

- On the desktop of the xp-virtual machine, you would find Rhadoop.zip. Copy this to the home folder of hduser on your server using scp.
- ssh into hduser of the server in which you had configured R in the previous step. Extract RHadoop.zip.

4.1 Installing Apache-thrift

- Run the following commands in bash terminal

```
cd ~/RHadoop/thrift-0.9.0/
sudo ./configure
sudo make
sudo make check
sudo make install
```

4.2 Installing Depencencies

- Open R with sudo privilidges (Command : `sudo R`).

4.2.1 Installing Using install.packages()

- Install the dependencies for RHadoop packages by running the following command in the Rconsole.

```
install.packages(c("rJava", "Rcpp", "RJSONIO", "
    ↪ bitops", "digest","functional","stringr", "plyr
    ↪ ", "reshape2", "dplyr","R.methodsS3", "caTools"
    ↪ , "Hmisc"))
install.packages("data.table")
```

You would be given a list of download servers sources to choose from. Select Vietnam or Russia. It may be possible that the selected server does not provide all the above package. I case of this or if you do not have a good internet connection use manual installation.

4.2.2 Manually Installing Dependencies

- Close R (if open and go to bash terminal)
- `cd /RHadoop/Dependencies/`
- Open R in sudo mode and type
  `source('Install_Manually.r')`.
- `This should manually install all the dependecies for RHadoop.`

4.3 Setting up RHadoop

- `Add the following lines to .bashrc in the home folder of hduser for setting the enviornment variable`

```
export HADOOP_OPTS="-Djava.library.path=
    ↪ $HADOOP_INSTALL/lib"
export HADOOP_CMD="/usr/local/hadoop/bin/hadoop"
export HADOOP_STREAMING="/usr/local/hadoop/share/
    ↪ hadoop/tools/lib/hadoop-streaming-2.4.1.jar"
export HADOOP_PREFIX="/usr/local/hadoop"
export LD_LIBRARY_PATH="/usr/lib/jvm/java-6-
    ↪ openjdk-i386/jre/lib/i386/ser"
```

- `cd /RHadoop/Dependencies/`
- `Open R in sudo mode and type source(ścript.r)́.`
- `This should install all the required libraries of RHadoop.`
- `Follow the exact same procedure and install R and RHadoop on each node of the hadoop cluster.`

5 Results

- The first thing we tried was to set up Ubuntu on the blade servers provided to us.

  A blade server is a stripped down server computer with a modular design optimized to minimize the use of physical space and energy. This caused a lot of problem initially for we found out that Ubuntu 14.04 causes problems on the blade servers, so Ubunutu 12.04 was installed after some problems.

- Four nodes were set up on four blade clusters - 3,4,7,8
- We installed Hadoop on two nodes (3 - master and 8 -slave8) and ran a sample code (calculating the value of pi) and were able to use Map/Reduce to do our work. The code worked perfectly.[5]
- We ran the code given on the next page in R to count the number of occurrences of different words in a 2GB text file.

```
Sys.setenv("HADOOP_CMD"="/usr/local/hadoop/bin/hadoop
    ↪ ")
Sys.setenv("HADOOP_STREAMING"="/usr/local/hadoop/
    ↪ share/hadoop/tools/lib/
hadoop-streaming-2.4.1.jar")
Sys.setenv("HADOOP_PREFIX"="/usr/local/hadoop")
Sys.setenv("LD_LIBRARY_PATH"="/usr/lib/jvm/java-6-
    ↪ openjdk-i386/jre/lib/i386/server")
library(rJava)
library(rhdfs)
hdfs.init()
library(rmr2)
```

```
## map function
map <- function(k,lines) {
words.list <- strsplit(lines, '\\s')
words <- unlist(words.list)
return( keyval(words, 1) )
}

## reduce function
reduce <- function(word, counts) {
keyval(word, sum(counts))
}

wordcount <- function (input, output=NULL) {
mapreduce(input=input, output=output, input.format="
    ↪ text",
map=map, reduce=reduce)
}


## delete previous result if any
system("/usr/local/hadoop/bin/hadoop dfs -rm -R /user
    ↪ /hduser/InputData4")
system("/usr/local/hadoop/bin/hadoop dfs -
    ↪ copyFromLocal /home/hduser/Input /user/hduser/
    ↪ InputData4")
system("/usr/local/hadoop/bin/hadoop dfs -rm -R /user
    ↪ /hduser/InputData4")

## Submit job
hdfs.data <- file.path('/user/hduser', 'InputData4')
hdfs.out <- file.path('/user/hduser', 'OutputData4')


web-based out <- wordcount(hdfs.data, hdfs.out)


results <- from.dfs(out)

## check top 30 frequent words
results.df <- as.data.frame(results, stringsAsFactors
    ↪ =F)
colnames(results.df) <- c('word', 'count')
v1=results.df[order(results.df$count, decreasing=T),
    ↪ ];
```

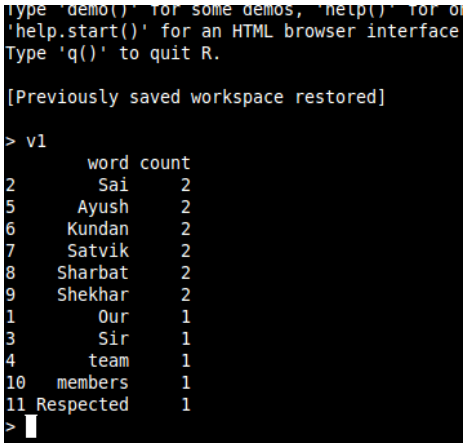


Figure 3: Output of wordcount for a small input text file

Number of nodes	Type of program	Size of input	Time taken
3	Wordcount	1.8GB	21m4s
2	Wordcount	1.8GB	27m33s
3	Terasort	1GB	2m47s
2	Terasort	1GB	5m3s

6 Future work

- We plan to build a gui interface through which you can upload your R-code and the system will run the code on the distributed servers and genereate an output file for you to download [6]
- This UI will be online which can be hosted internally or made public to outsiders also to submit jobs.
- User will be notified by email when the output file has been generated
- We intend to create an installation package that given the ip addresses of the master and slave nodes(which have ubuntu installed in them) will set up hadoop on all those nodes and integrate them.
- We plan to have more slave nodes to the cluster and ensure a better node distribution.

References

[1] Revolution Analytics. Rhadoop. https://github.com/RevolutionAnalytics/RHadoop, 2011.

[2] Apache Software Foundation. Apache hadoop, 2014.

[3] Michael Noll.

[4] Yahoo Software. Yahoo! hadoop tutorial. 2011.

[5] Yanchang Zao. Hadoop: from single-node mode to cluster mode. *RDataMining.com*, 2013.

[6] Yanchang Zao. Step-by-step guide to setting up an r-hadoop system. *RDataMining.com*, 2014.