Lab Assignment and Lab Manual CSE 408

**Distributed Systems CSE 408**

Lab 1: Hadoop

**Setup**

You may work on this lab with a partner or individually. If you aren’t confident in your Java  skills, we strongly recommend that you find a partner with good Java skills.

Required Software

Required software for Linux and Windows include:

1. JavaTM 1.6.x, preferably from Sun, must be installed.
2. **ssh** must be installed and **sshd** must be running to use the Hadoop scripts that manage remote Hadoop daemons.

Installing Software

If your cluster doesn't have the requisite software you will need to install it. software is copied on your machine may be in a folder named DS or soft. hadoop-0.20.203.0.tar.gz

Unpack the downloaded available  Hadoop distribution. using

**$ tar zxvf hadoop-0.20.203.0.tar.gz**

Hadoop folder will be created  Go to that directory and

Try the following command:  
**$ bin/hadoop**  
This will display the usage documentation for the **hadoop** script.

Now you are ready to start your Hadoop cluster in one of the three supported modes:

* Local (Standalone) Mode
* Pseudo-Distributed Mode
* Fully-Distributed Mode

Define at least JAVA\_HOME  and PATH by typping the following command

**$export JAVA\_HOME=/home/studcse/jdk\_installation path**

**$export PATH=$JAVA\_HOME/bin:$PATH**

Test which java you are using by the following command

**$ which java**

**/home/studcse/jdk\_installtion\_path/bin/java**

**Standalone Operation of Hadoop**

By default, Hadoop is configured to run in a non-distributed mode, as a single Java process. This is useful for debugging. The following example copies the unpacked conf directory to use as input and then finds and displays every match of the given regular expression. Output is written to the given output directory.

**$ mkdir input  
$ cp conf/\*.xml input  
$ bin/hadoop jar hadoop-examples-\*.jar grep input output 'dfs[a-z.]+'   
$ cat output/\***

**Pseudo-Distributed Operation**

Hadoop can also be run on a single-node in a pseudo-distributed mode where each Hadoop daemon runs in a separate Java process.

**Configuration**

Use the following:

conf/core-site.xml:

**<configuration>**

**<property>**

**<name>fs.default.name</name>**

**<value>hdfs://localhost:9000</value>**

**</property>**

**</configuration>**

**conf/hdfs-site.xml:**

**<configuration>**

**<property>**

**<name>dfs.replication</name>**

**<value>1</value>**

**</property>**

**</configuration>**

**conf/mapred-site.xml:**

**<configuration>**

**<property>**

**<name>mapred.job.tracker</name>**

**<value>localhost:9001</value>**

**</property>**

**</configuration>**

Setup passphraseless ssh

Now check that you can ssh to the localhost without a passphrase:  
**$ ssh localhost**

If you cannot ssh to localhost without a passphrase, execute the following commands:  
**$ ssh-keygen -t dsa -P '' -f ~/.ssh/id\_dsa  
$ cat ~/.ssh/id\_dsa.pub >> ~/.ssh/authorized\_keys**

Execution

Format a new distributed-filesystem:  
**$ bin/hadoop namenode -format**

Start the hadoop daemons:  
**$ bin/start-all.sh**

The hadoop daemon log output is written to the ${HADOOP\_LOG\_DIR} directory (defaults to ${HADOOP\_HOME}/logs).

Browse the web interface for the NameNode and the JobTracker; by default they are available at:

* NameNode - <http://localhost:50070/>
* JobTracker - <http://localhost:50030/>

Copy the input files into the distributed filesystem:  
**$ bin/hadoop fs -put conf input**

Run some of the examples provided:  
**$ bin/hadoop jar hadoop-examples-\*.jar grep input output 'dfs[a-z.]+'**

Examine the output files:

Copy the output files from the distributed filesystem to the local filesytem and examine them:  
**$ bin/hadoop fs -get output output  
$ cat output/\***

or

View the output files on the distributed filesystem:  
**$ bin/hadoop fs -cat output/\***

When you're done, stop the daemons with:  
**$ bin/stop-all.sh**

Hadoop Command

hadoop command [genericOptions] [commandOptions]

Examples:-

command – fs, jar, job

[genericOptions] - -conf, -D, -files, -libjars, -archives

[commandOptions] - -ls, -submit

HDFS Commands [1]

$ hadoop fs –ls

If you see an error do the following where

[username] is your training account username

$ hadoop fs -mkdir /user/[username]

$ vi testfile1 [ Repeat for testfile2]

This is file 1

This is to test HDFS

$ hadoop fs -mkdir input

$ hadoop fs -put testfile\* input

You can get help on commands -

$ hadoop fs -help

$ hadoop fs -cat input/testfile1

$ hadoop fs -cat input/testfile\*

Download the files from HDFS into a directory called

input and check there is a input directory.

$ hadoop fs -get input input

$ ls input/

$ hadoop job -list

Input in HDFS

$ hadoop fs -mkdir wordcount-in

$ hadoop fs -put /global/scratch/sd/lavanya/

hadooptutorial/wordcount/\* wordcount-in/

Run example

$ hadoop jar /usr/common/tig/hadoop/

hadoop-0.20.2+228/hadoop-0.20.2+228-examples.jar

wordcount wordcount-in wordcount-op

View output

$ hadoop fs -ls wordcount-op

$ hadoop fs -cat wordcount-op/part-r-00000

$ hadoop fs -cat wordcount-op/p\* | grep Darcy

Wordcount: Number of reduces

$ hadoop dfs -rmr wordcount-op

$ hadoop jar /usr/common/tig/hadoop/

hadoop-0.20.2+228/hadoop-0.20.2+228-examples.jar

wordcount -Dmapred.reduce.tasks=4 wordcount-in

wordcount-op

Streaming with Unix Commands

$ hadoop jar $HADOOP\_HOME/contrib/streaming/

hadoop\*-streaming.jar -input wordcount-in -output

wordcount-streaming-op -mapper /bin/cat -reducer /

usr/bin/wc

$ hadoop fs -cat wordcount-streaming-op/p\*