Hadoop Configuration for Processing Full Bitcoin Transaction Dataset
(single node pseudo-cluster)

Launch Instance
open http://aws.amazon.com and login
launch instance: ubuntu 64-bit paravirtualized
instance type: m3.xlarge
storage: 100 GB (5 to 10 times the full data size)
security group: ALL TCP 0.0.0.0/0 (not safe, but simple testing)

Connect to server using SSH
(recommended client “Bitvise SSH Client”)

```
sudo apt-get update
sudo apt-get install openjdk-7-jdk
```

Add hadoop user and configure account

```
sudo -s
useradd -d /home/hadoop -m hadoop
passwd hadoop
(set password to: h)
usermod -a -G sudo hadoop
usermod -s /bin/bash hadoop
su hadoop
cd ~
```

Configure SSH keys

```
ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys

sudo chmod go-w $HOME $HOME/.ssh
sudo chmod 600 $HOME/.ssh/authorized_keys
sudo chown `whoami` $HOME/.ssh/authorized_keys
```

Update linux equivalent of Windows path variables

```
nano ~/.bashrc
export HADOOP_PREFIX="/home/hadoop"
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
```

Load path variables
source ~/.bashrc

Install AWS tools and configure
sudo apt-get install python-pip
sudo pip install awscli

sudo aws configure
<access key id from email>
<access key from email>

Download hadoop and extract to hadoop user home folder
wget http://web.njit.edu/~jbc8/download/hadoop-2.2.0.tar.gz
tar -zxvf ~/hadoop-2.2.0.tar.gz
mv ~/hadoop-2.2.0/* ~
rm ~/hadoop-2.2.0
rm ~/hadoop-2.2.0.tar.gz

Update hadoop config files
(why does hadoop not come these as the default configuration?)

nano ~/etc/hadoop/core-site.xml

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:8020</value>
    <description>The name of the default file system.  Either the
      literal string "local" or a host:port for HDFS.
    </description>
    <final>true</final>
  </property>
</configuration>

<configuration>
  <property>
    <name>dfs.namenode.name.dir</name>
    <value>file:/home/hadoop/data/namenode</value>
    <description>Determines where on the local filesystem the DFS name node
        should store the name table.  If this is a comma-delimited list
        of directories then the name table is replicated in all of the
        directories, for redundancy.
```

```
    </description>
    <final>true</final>
  </property>
  <property>
    <name>dfs.datanode.data.dir</name>
    <value>file:/home/hadoop/data/datanode</value>
    <description>Determines where on the local filesystem an DFS data node
       should store its blocks.  If this is a comma-delimited
       list of directories, then data will be stored in all named
       directories, typically on different devices.
       Directories that do not exist are ignored.
    </description>
    <final>true</final>
    </property>
    <property>
      <name>dfs.replication</name>
      <value>1</value>
    </property>
    <property>
    <name>dfs.permissions</name>
    <value>false</value>
  </property>
</configuration>

mkdir ~/data ~/data/namenode ~/data/datenode
```

Increase memory limit in JVM
(memory limit for m3.large is 15 GB)

```
cp ~/etc/hadoop/mapred-site.xml.template etc/hadoop/mapred-site.xml
nano ~/etc/hadoop/mapred-site.xml

<configuration>
  <property>
    <name>mapreduce.framework.name</name>
    <value>yarn</value>
  </property>

  <property>
    <name>mapred.map.java.opts</name>
    <value>-Xmx2048m</value>
  </property>
</configuration>
```

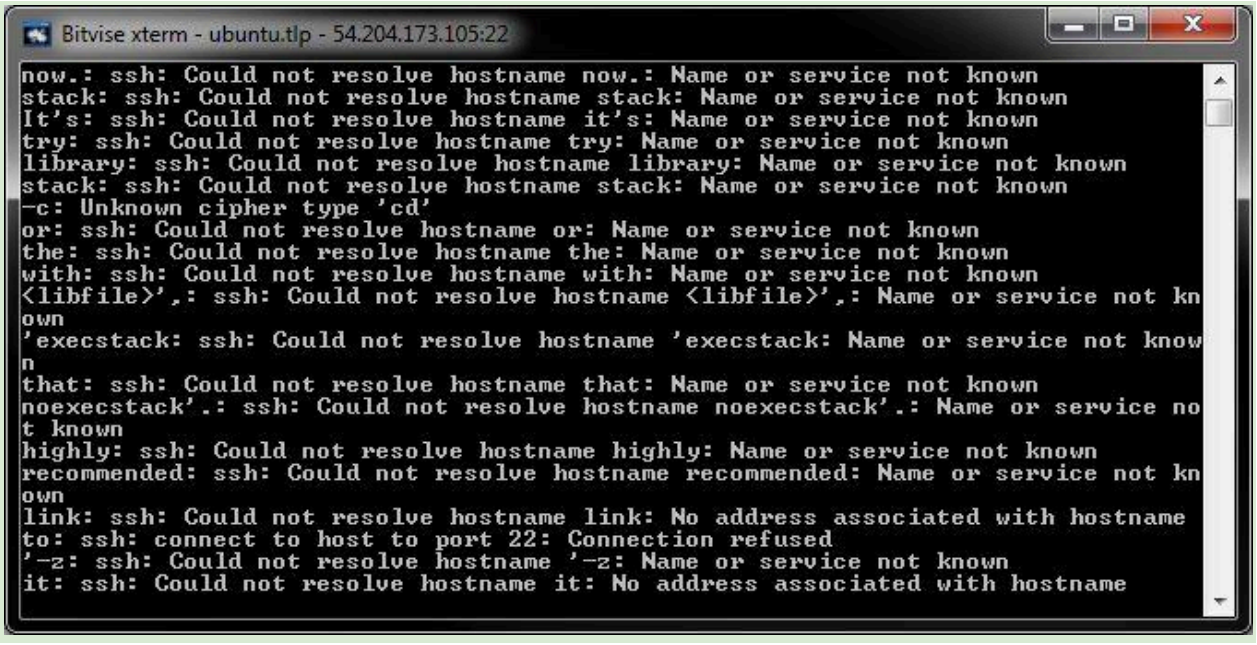
```
  <property>
    <name>mapred.reduce.java.opts</name>
    <value>-Xmx2048m</value>
  </property>
</configuration>
```

nano ~/etc/hadoop/yarn-site.xml

```
<configuration>
<!-- Site specific YARN configuration properties -->
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
  <property>
    <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
    <value>org.apache.hadoop.mapred.ShuffleHandler</value>
  </property>
</configuration>
```

nano ~/etc/hadoop/hadoop-env.sh
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64

It's OK to ignore all the ssh errors, 64-bit warnings, and strange text. The important part is that all the services are running, check with: jps



hdfs namenode -format
start-all.sh
mr-jobhistory-daemon.sh start historyserver
jps

Download input data
sudo aws s3 cp s3://group-haifa-amble-jesse/sorted.csv ~

hadoop dfsadmin -safemode leave
hdfs dfs -mkdir /input
hdfs dfs -copyFromLocal ~/sorted.csv /input
wget http://web.njit.edu/~jbc8/download/BitcoinHadoopComplex.jar ~

hadoop jar BitcoinHadoopComplex.jar /input output

You can check disk and memory usage as the program runs with the following commands:
df -h
(disk space free)

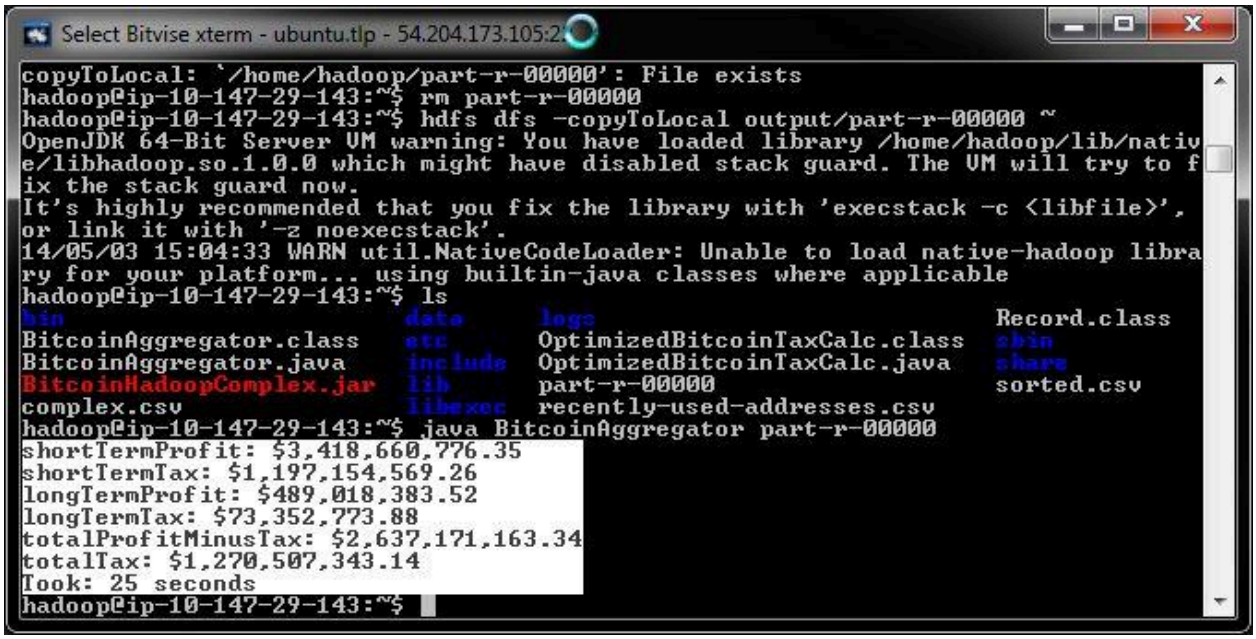
free
(memory usage)

hdfs dfs -tail output/part-r-00000
hdfs dfs -copyToLocal output/part-r-00000 ~
aws s3 cp part-r-00000 s3://group-haifa-amble-jesse/

(if you need to rerun the MapReduce program, remove output using)
hdfs dfs -rmr output

wget http://web.njit.edu/~jbc8/download/BitcoinAggregator.java
javac BitcoinAggregator.java
java BitcoinAggregator part-r-00000

Expected results
shortTermProfit: $3,418,660,776.35
shortTermTax: $1,197,154,569.26
longTermProfit: $489,018,383.52
longTermTax: $73,352,773.88
totalProfitMinusTax: $2,637,171,163.34
totalTax: $1,270,507,343.14
Took: 25 seconds



Source code
http://web.njit.edu/~jbc8/download/BitcoinMapReduceTax.java
(requires:
commons-cli-1.2.jar,
commons-logging-1.1.1.jar,
hadoop-common-2.2.0.jar,
hadoop-mapreduce-client-core-2.2.0.jar)

http://web.njit.edu/~jbc8/download/BitcoinAggregator.java

If your IP address changes from a stop/start and SSH stops working
rm ~/.ssh/known_hosts