Program Report

**Description**

* Hadoop Cluster Setup Single Node and 16 Nodes setup-
  + Launch spot instance c3.large with required settings
  + Download the key and copy the key to instance.
  + Setup password less ssh for the connection, so that once you have many slaves, ssh wont ask you to provide any password.
    - eval `ssh-agent -s`
    - ssh-add <keyname.pem>
    - chmod 600 <keyname.pem>
  + Download Hadoop – wget <http://apache.mirror.gtcomm.net/hadoop/common/hadoop-2.7.2/hadoop-2.7.2.tar.gz>
  + Extract Hadoop – tar -xzvf hadoop-2.7.2.tar.gz
  + Move hadoop to new folder - mv hadoop-1.2.1 hadoop
  + vi .bashrc to set following paths
    - export CONF=/home/ubuntu/hadoop/etc/hadoop
    - export JAVA\_HOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64
    - export PATH=$PATH:$JAVA\_HOME/bin
    - export HADOOP\_HOME=/home/ubuntu/hadoop
    - export PATH=$PATH:$HADOOP\_HOME/bin
  + Go to in hadoop/etc/hadoop/ and change the following configuration files-
    - Core-site.xml – Hadoop configuration file for setting up namenode. The namenode is basically the master node, we can consider it as a current instance. Add following settings in the file.
      * <property>

<name>fs.defaultFS</name>

<value>hdfs://ec2-52-48-64-222.eu-west-1.compute.amazonaws.com:9000</value>

</property>

* hdfs-site.xml – configurations related to hdfs site. Like for restricting replication of the file system, set it to 1. By default Hadoop gives replication as 3. Turning off the permissions, set it to false Please check the hdfs-site.xml file for the changes made.
* Mapred-site.xml – Here we mention the host and port that the mapreduce job tracker runs at. It also includes the yarn framework. The idea of the yarn is to split up the tasks of resource manager and monitor or schedule jobs into separate processes/daemons.
* Yarn-site.xml – As discussed above the idea of yarn framework, we need to make some changes to the configuration files. Like provider the resource manager IP address, the scheduler address, the webapp address, the resource tracker address and the resource admin address.

**What is a Master node? What is a Slaves node?**

* Masters.xml – Masters.xml will have details of the master or namenode instance. It will handle operations like opening, closing and renaming file and directories and determines the mapping of blocks to datanodes along with regulating access to files by clients. As it will the launching point of the app. When configuring for multiple nodes, we need to make sure that the data nodes masters.xml is empty.
* Slaves.xml – Basically represent the datanode which executes tasks upon instruction from the master and also handles the data between map and reduce phases. Every slave file will have it corresponding public dns, so that the master can connect to the file. The masters slave.xml will have one or all dns names of slaves, depending upon how much clusters are attached.

**Why do we need to set unique available ports to those configuration files on a shared environment? What errors or side-effects will show if we use same port number for each user?**

* Hadoop has its own ports set for connecting the cluster through web-UI or through namenode. The namenode port is provided by Inter Process communicator port which is standardly defined. If every uses the same nodes, there will be port busy errors. Although, Hadoop will provide different ports even if the ports are busy.

**How can we change the number of mappers and reducers from the configuration file?**

* You can add the config mapred.map.tasks = <any number> of mappers you want to use.
* You can add the config mapred.reducer.tasks =<number of reducers you want>.
* Both these changes an be added to mapred-site.xml
* After all the above changes, navigate to the bin folder, and format the hdfs file system, using following commands
  + - ./hdfs namenode –format
* Start the file system -> ./start-dfs.sh
* Start the yarn -> ./start-yarn.sh resource and node managers will start too

Jps -> will show the current active nodes.

The above installation will start Hadoop and we can deploy our application and execute it.

**Spark Installation-**

* Install Hadoop as above and start the the nodes.
* Download spark with Hadoop
* Save and extract and move it to spark folder.
* In spark/conf make some config changes.
  + Make changes to file slaves.xml, add number of slaves to connect.
  + Similarily change the file spark-env.sh and add export SPARK\_PUBLIC\_DNS=”public dns of your instance”
* Download scala and set the path as follows for both spark and scala
  + - export SCALA\_HOME=/home/ubuntu/scala
    - export PATH=$PATH:$SCALA\_HOME/bin
    - export SPARK\_HOME=/home/ubuntu/spark
    - export PATH=$PATH:$SPARK\_HOME/bin
* run spark-shell command and check whether spark is installed.
* Now run ./start-all.sh from spark/sbin directory
* Jps and check, the directory, the master and slaves will start.

**System Details-**

Instance – c3.large

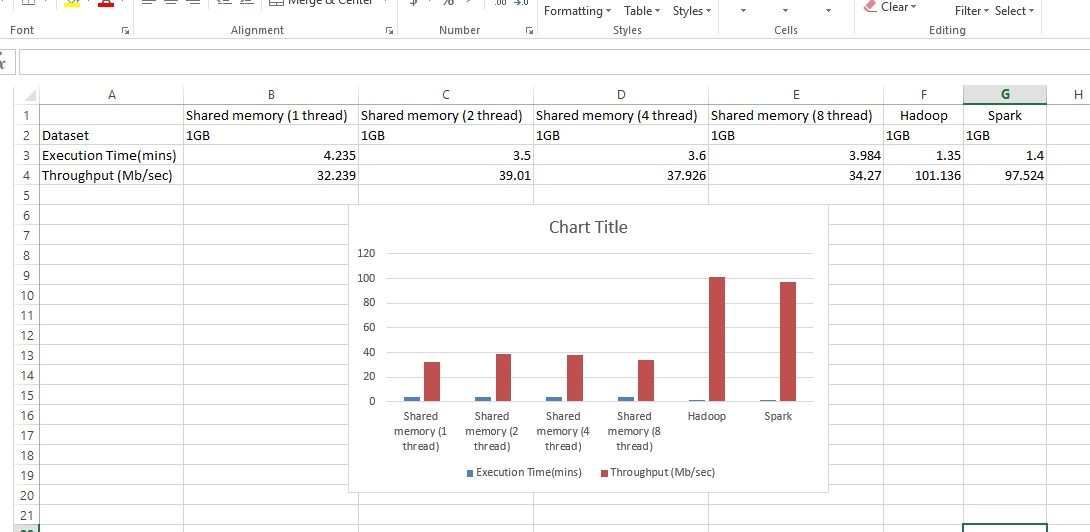
Region – eu-west-1 (Europe-Ireland)

Java version – java-1.7.0-openjdk

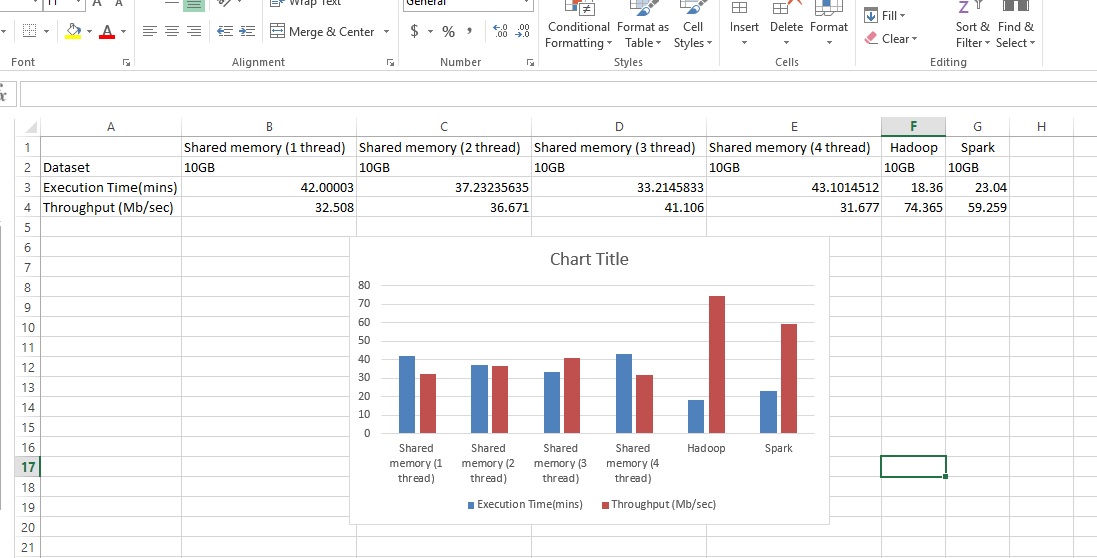
Maven Version – Apache Maven 3.0.5

**Execution Time**

**1GB Graph**



**10GB Graph**

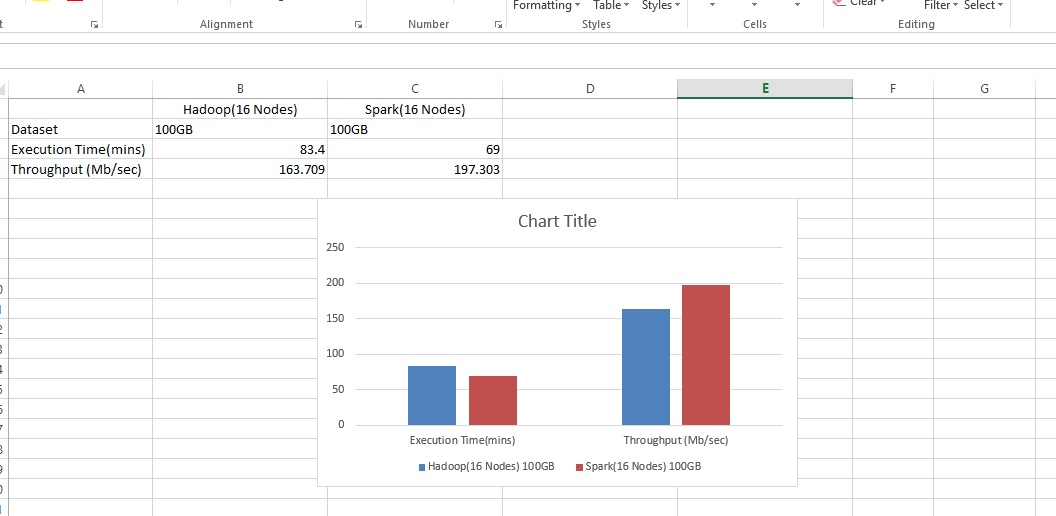


Explanation- On the basis of above graphs for 1GB and 10GB file sizes with 1 Nodes, it can be identified that Hadoop and Spark provide faster processing of data than Shared Memory sort.

Hadoop and Spark both frameworks where developed to handle large datasets on multiple nodes, using the idea of distributed computing.

Hence the above difference between execution time and throughput of Shared Memory vs Hadoop and Spark is higher for latter frameworks.

**100GB Graph-**



Explanation - The above dataset shows that spark has a better execution rate because Spark is an alternative to the traditional approach that can be used for real time stream data processing.

It uses the resilient data sets to reproduce data across all the nodes.

Hence, in the above graph we observe a major difference between executions between Hadoop and Spark 16 Nodes.