Aditya Kale – A20355807

Cloud Computing

Programming Assignment – 2

- **Problem**
  - In this experiment we have sorted 10GB and 100GB of text file data using a different sorting program.
  - File size which are greater than memory are sorted in this experiment.
  - Results of all data are compared among themselves and graph are plotted. Graphs are used for comparing results.

- **Methodology:**
  - Different implemented sorting logic is used for sorting large files.
  - Our own code is evaluated for files having size greater than memory.
  - Hadoop Map Reduce and Spark sorting is used to sort files.

- **Operating System Used:**
  - Linux ubuntu 14.04
  - Linux kernel version – 3.19.0-25-generic

- **Versions:**
  - Ant-1.9.6
  - Java-8
  - Hadoop-2.7.2
  - Spark – 1.6.0

- **Configuration files description.**
  - **etc/hosts**
    - This file has all IP addresses and DNS of master and all slaves in an Hadoop cluster.

    Ex: **172.31.15.201 ec2-52-38-119-195.us-west-2.compute.amazonaws.com**
    172.31.15.202 ec2-52-38-119-196.us-west-2.compute.amazonaws.com

  - **conf/slaves**
    - This file has all DNS value of Master node and all Slave nodes in an Hadoop cluster.
      Ex: **ec2-52-38-119-195.us-west-2.compute.amazonaws.com**

  - **conf/core-site.xml**
    - This file is used to configure parameters for specifying NameNode URI, Hadoop temporary directory etc.
    - **Parameter** : fs.default.name
      Value : URI of NameNode (Master Node dns:port).

    - **Parameter :** haoop.tmp.dir
      value : base location for other hdfs directories

  - **conf/hdfs-site.xml**
    - This file can be used to configure DFS directory, replication factor and other properties.
    - **Parameter** : dfs.name.dir
      value : Path on the local filesystem where the Name Node stores namespace
    - **Parameter** : dfs.data.dir
      Value : It's a path on the local filesystem of a Data Node where it should store its blocks
    - **Parameter** : dfs.replication
      Value : Its specify the number of replicas of data to be generated.

  - **conf/mapred-site.xml**
    - This file is used to configure the mapper reducer tasks, to specify map-reduce framework name etc.
    - **Parameter** : mapred.job.tracker
      Value: Host or IP and port of JobTracker.
    - **Parameter** : mapreduce.framework.name
      Value: It specify the mapreduce framework name (yarn).
    - **Parameter** : mapred.tasktracker.map.tasks.maximum
      Value : No of The maximum number of Mapper tasks, which are run simultaneously on a given TaskTracker
    - **Parameter** : mapred.tasktracker.reduce.tasks.maximum
      Value : The maximum number of Reducer tasks, which are run simultaneously on a given TaskTracker

  - **conf/yarn-site.xml**
    - In this file shuffler class can be configure, resource managers web address can be specified which tells on which address we could track the progress. Also

    - **Parameter** : yarn.nodemanager.aux-services.mapreduce.shuffle.class
      Value: org.apache.hadoop.mapred.ShuffleHandler
    - **Parameter** : yarn.resourcemanager.webapp.address
      Value: "Master_DNS":"Port"

- **Installation Steps to setup Virtual Cluster.**
  - Installation steps to setup virtual cluster of 16 nodes Hadoop and Spark is given in ReadMe file.

- **Modification required to go from 1 node to 16 node.**

  Stop all data node and namenode using
  hadoop/bin$ ./stop-all.sh

  Format Namenode:
  hadoop/bin$ bin/hadoop namenode -format

  Add all ip addresses entries in
  /etc/hosts

  Add all DNS into slaves files
  /hadoop/etc/hadoop/slaves file

  Evaluate ssh-agent
  eval "$(ssh-agent)"

  Give Permission to pem file
  chmod 600 aditya1.pem

  ssh-add aditya1.pem

  Generate RSA keys using below command. RSA keys are used for communication between master and different slave nodes.
  ssh-keygen -t rsa

  Below Command copies slave's RSA id to Master's authorized keys. Which allows Master node to communicate with slave nodes.

  ssh-copy-id -i ~/.ssh/id_rsa.pub ubuntu@"MASTER_DNS"

  Change permission of Master's authorized keys.
  chmod 0600 ~/.ssh/authorized_keys

  Now start dfs and yarn to up Data Node and Name node, resource manager.
  hadoop/bin$ ./start.dfs.sh
  hadoop/bin$ ./start.yarn.sh

  Check whether DataNode, NameNode, ResourceManager is been up using following command
  hadoop/bin$ JPS

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

  - **Master**: It's a main node where task are executed. Tasks are given to Master nodes take care of following key operations that comprise Hadoop:
    - Storing data in the Hadoop Distributed File System (HDFS)
    - Running parallel computations on that data using MapReduce.
    - The NameNode coordinates the data storage function (with the HDFS).
    - On the other hand JobTracker checks and coordinates the parallel processing of data.

    NameNode: Manages HDFS storage.

    Resource Manager: Oversees the scheduling of application tasks and management of the Hadoop cluster's resources.

  - Slaves:
    - The rest of the machines in a cluster act as both a DataNode and TaskTracker and are referred to as *slaves*.
    - Slave nodes make up the majority of virtual machines and perform the job of storing the data and running the computations.
    - Each worker node runs both a DataNode and TaskTracker service that communicates with, and receives instructions from their master nodes.
    - The TaskTracker service is subordinate to the JobTracker, and the DataNode service is subordinate to the NameNode.

- **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?**
  - NameNode : 8020
    JobTracker in case of MR1: 8021
    YARN Resource manager for MR2: 8032
    DataNodes: 50010 for non-secured and 1004 for secured clusters

  - In Hadoop architecture, URL: localhost: 50070 provides Web UI, which helps to understand the status of cluster, job details through browser. This helps, as user don't need to remember all commands from terminal. Other ports such as localhost: 50075 used for viewing datanode details, localhost: 8088 for viewing currently running jobs, completed jobs and so on.

  - Hadoop monitors lots of ports, also there are lots of hadoop daemons. Each monitors ports which can be override for specific reasons. YARN in hadoop has a port that monitors for job submission, yarn.resourcemanager.address.

- It is preferable to select different multicast addresses and ports for different distributed systems. Some operating systems may not synchronize separate between systems that uses unique addresses and by using same port number.

- If we use same number of ports conflict may occur between different services listening on same node.

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

- Number of mappers and reducer can be configure by adding properties in mapred-site.xml
- Adding following properties will change the number of mapper and reducer.

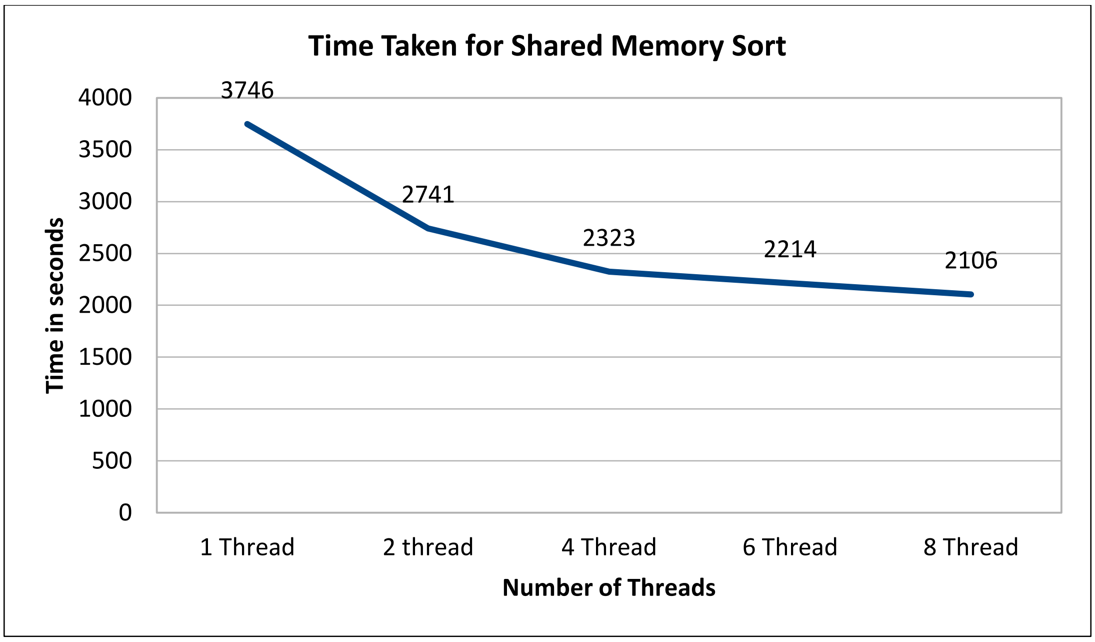
```
<property>
        <name>mapred.tasktracker.map.tasks.maximum</name>
        <value>4</value>
</property>
<property>
        <name>mapred.tasktracker.reduce.tasks.maximum</name>
        <value>4</value>
</property>
```

➢ **Results And Graphs**

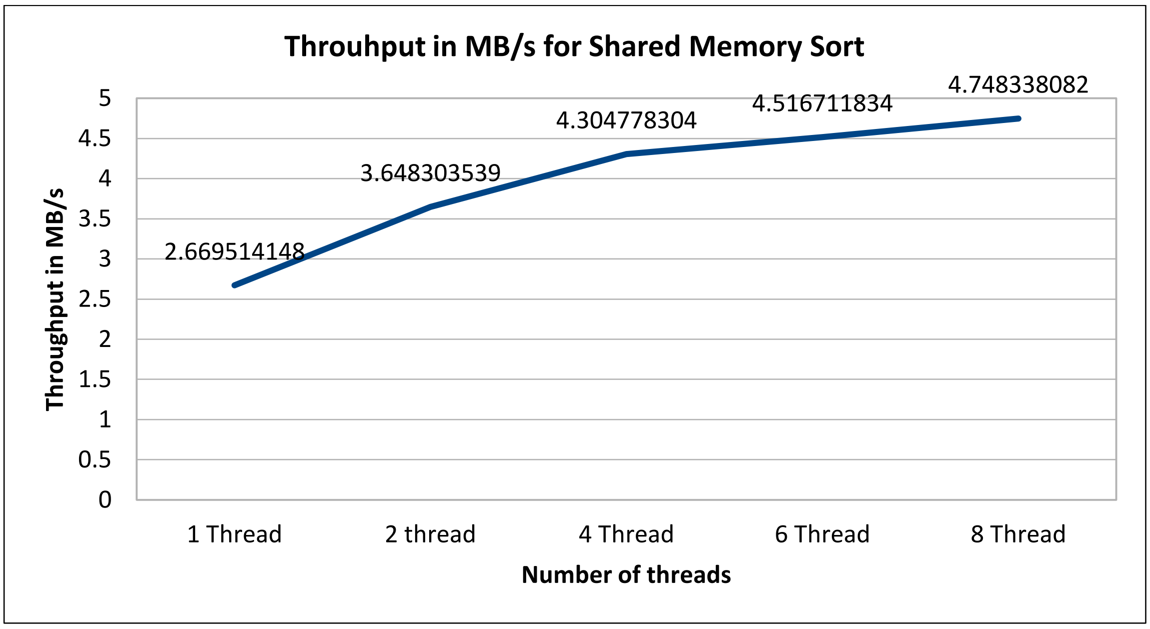
For Shared Memory Sort:

- This experiment is performed by **increasing** the **number of threads** such as **1,2,4,6,8.**
- **Fixed file size of 10GB** is used while performing this experiment
- Following results are captured by changing the number of thread value.
- And graphs are plotted based on these values.

	Time in seconds				
No of thread	1 Thread	2 Thread	4 Thread	6 Thread	8 Thread
	3746	2741	2323	2214	2106



	Throughput				
Throughput(MB/s)	1 Thread	2 thread	4 Thread	6 Thread	8 Thread
	2.66951	3.6483	4.30478	4.51671	4.74834



Conclusion:

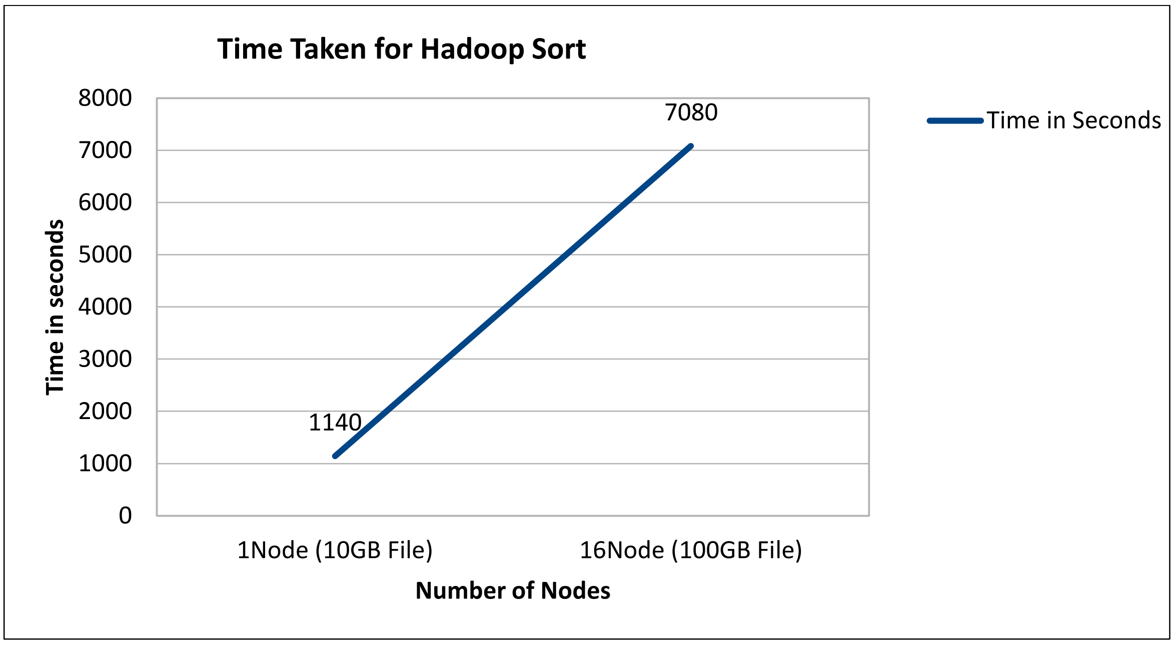
- As number of threads increases time required for sorting file decreases.
- As number of threads increases throughput of a system also increases.
- Throughput increases by a large amount when number of threads are changed form 1 thread to 4 threads. However after 4 threads throughput increases by very small amount.

- Also if further number of threads are increased throughput of a system will decrease as thread context switching will take more time.

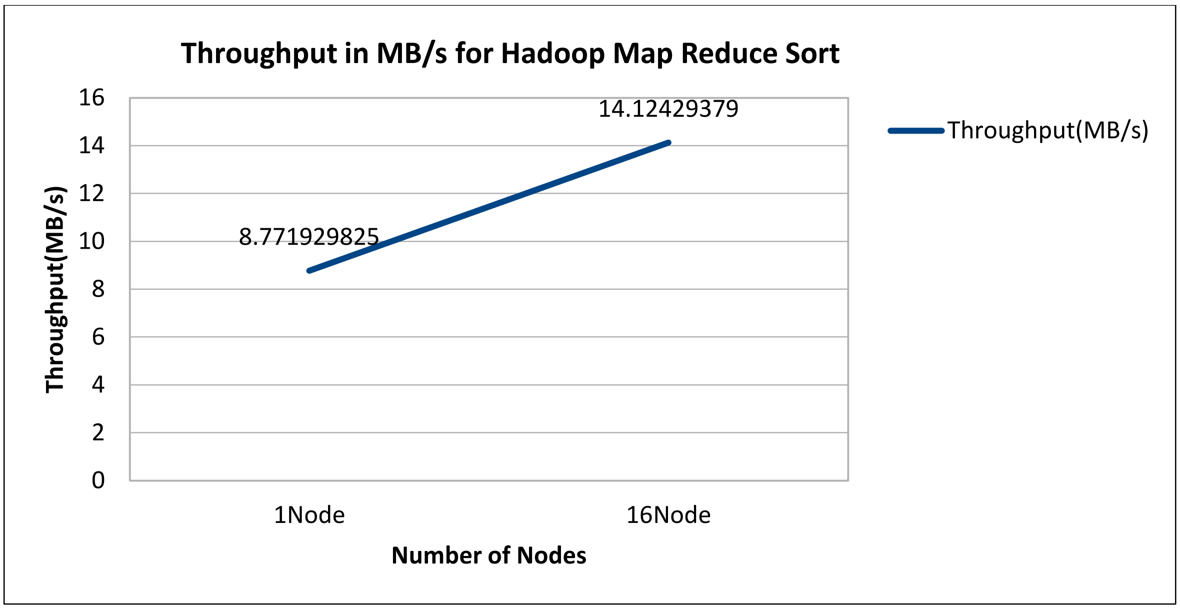
For Hadoop Map Reduce Sort:

- This experiment is performed **on 1 node and 16 node scale.**
- **File size is changed for 1 node and 16 node.**
- **10GB** is used for 1 node while 100GB file is used for 16 node while performing this experiment
- Following results are captured and graphs are plotted based on these values.

	Hadoop	Time in Seconds
Time in Seconds	1Node (10GB File)	1140
	16Node (100GB File)	7080



	Hadoop	Throughput(MB/s)
Throughput in MB/s	1Node	8.77192982
	16Node	14.1242938



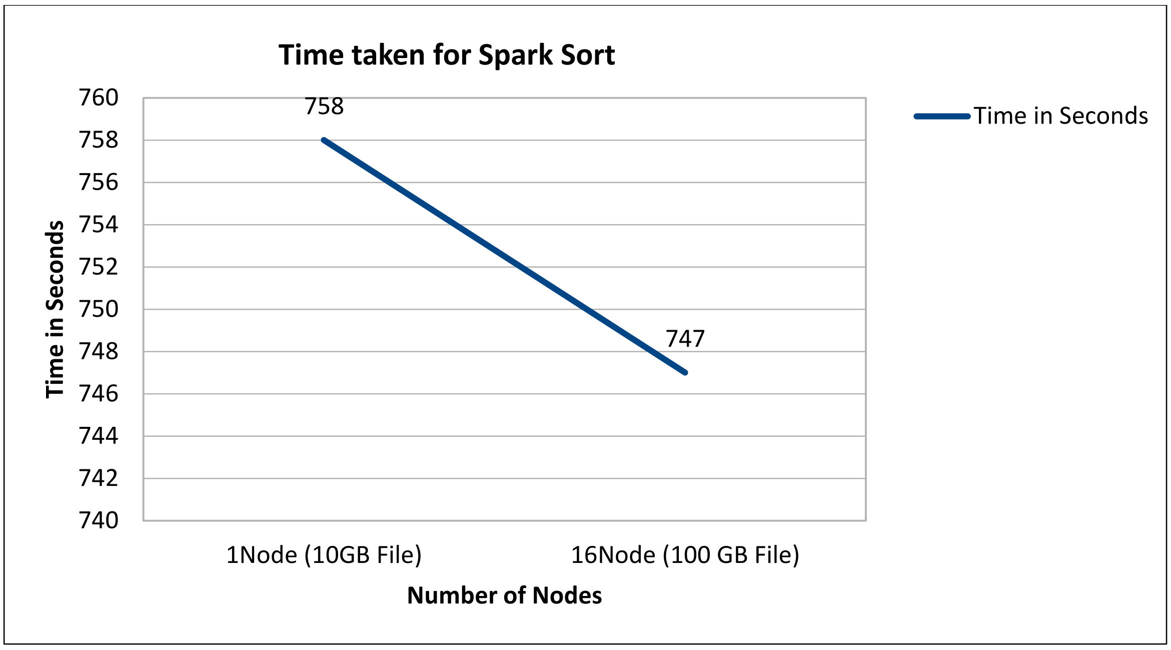
Conclusion:

- As number of nodes increased time required for sorting file decreases.
- As number of nodes increases throughput of a system also increases.
- As we increase the number of nodes parallel execution on data is done. Multiple nodes will perform the Mapper and reducer task are performed simultaneously, thus increasing the throughput.

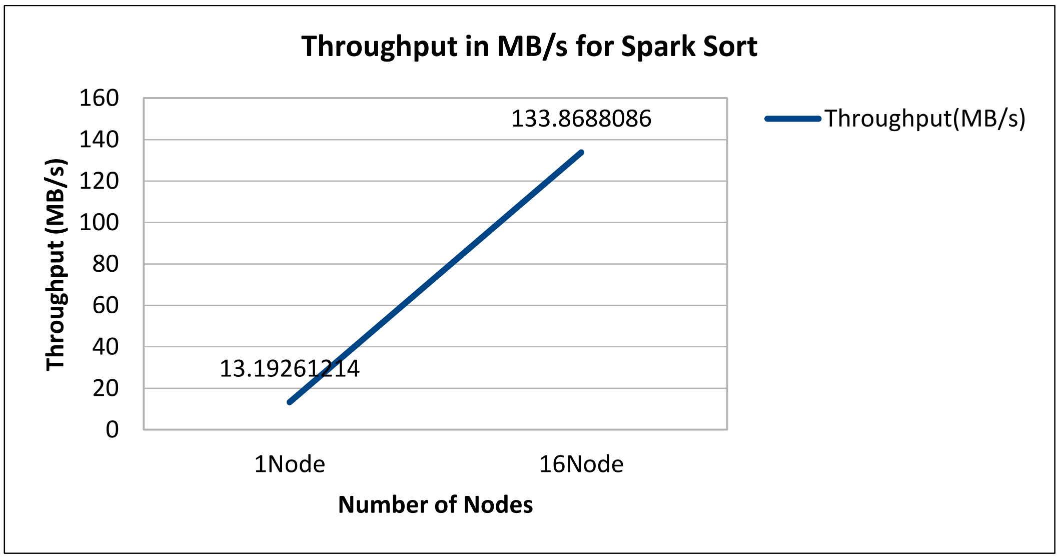
For Spark Sort:

- This experiment is performed **on 1 node and 16 node scale.**
- **File size is changed for 1 node and 16 node.**
- **10GB** is used for 1 node while 100GB file is used for 16 node while performing this experiment
- Following results are captured and graphs are plotted based on these values.

	Spark	Time in Seconds
Time in Seconds	1Node (10GB File)	758
	16Node (100 GB File)	747



	Spark	Throughput(MB/s)
Throughput in MB/s	1Node	13.19261214
	16Node	133.8688086

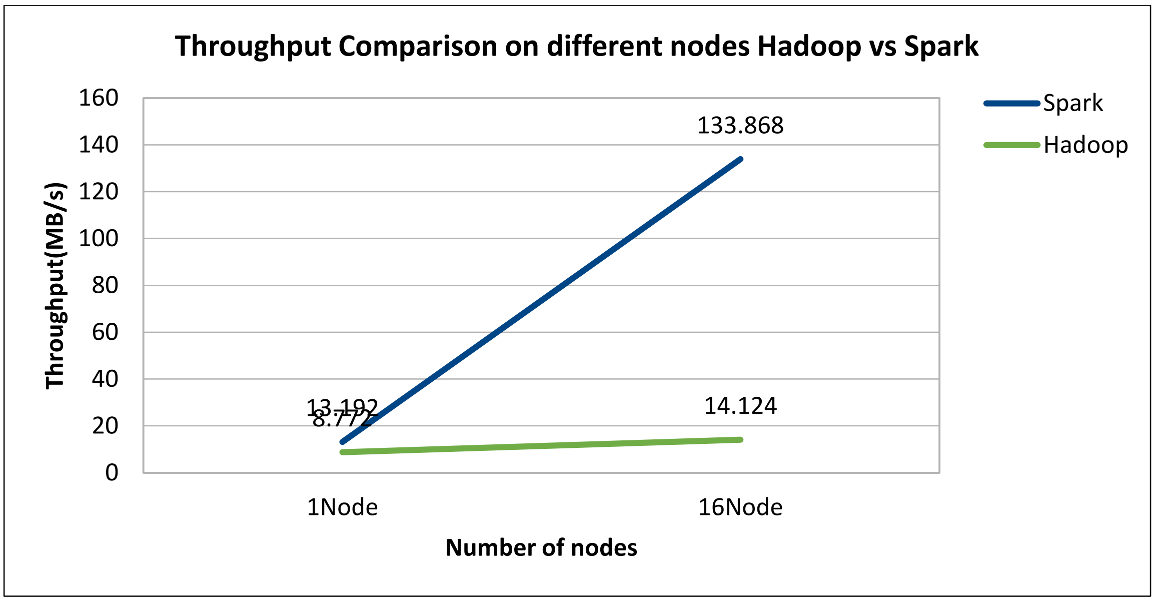


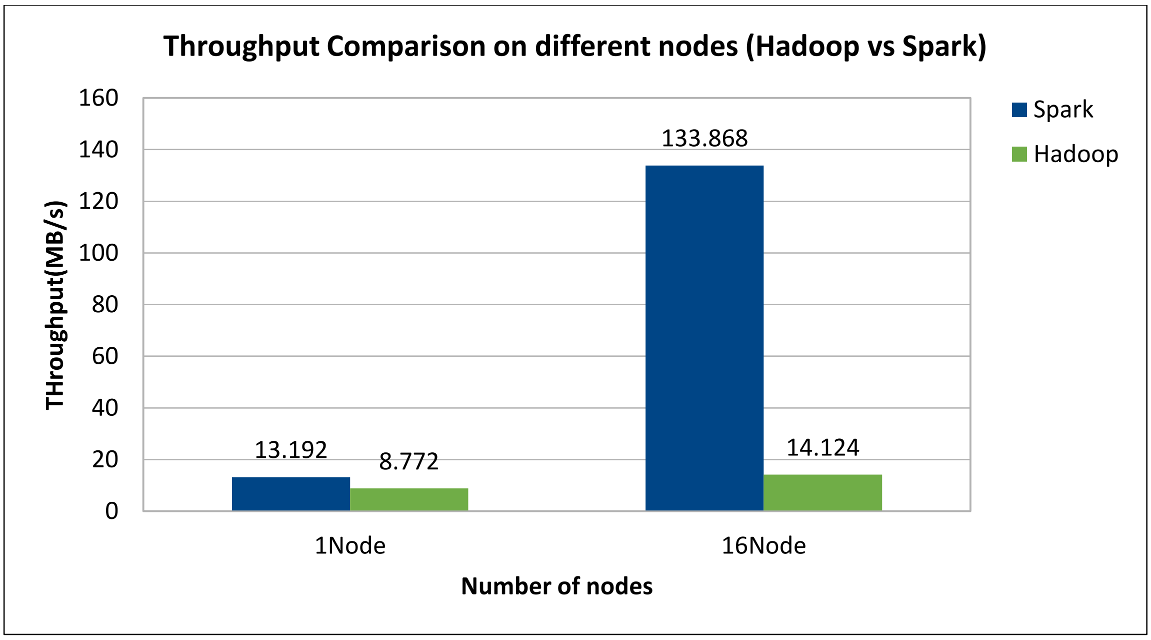
Conclusion:

- As number of nodes increased time required for sorting file decreases.
- Throughput of a system increases tremendously when jumped from 1 node to 16 node for Spark.

Comparing Hadoop vs Spark

	Spark	Hadoop
1Node	13.192	8.772
16Node	133.868	14.124





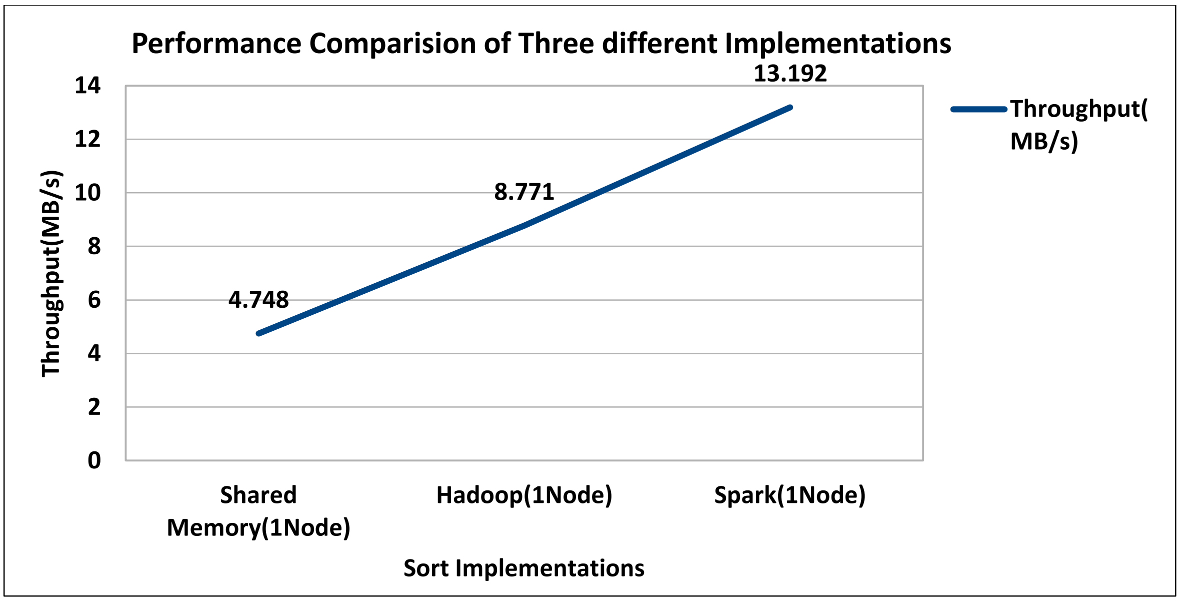
Conclusion:

- Comparing the above values we can say that Spark performs better than Hadoop in multi node cluster.
- Throughput of a Spark system increases by huge amount as number of nodes increases for large file size.

- **Comparing Shared Memory Sort Performance: with 1 and 16 nodes Hadoop and Spark.**

Comparing Performance of Shared Memory on 1 node with 1 node Hadoop and Spark.

Performance of Three different Sort Implementations	
	Throughput(MB/s)
Shared Memory(1Node)	4.748
Hadoop(1Node)	8.771
Spark(1Node)	13.192

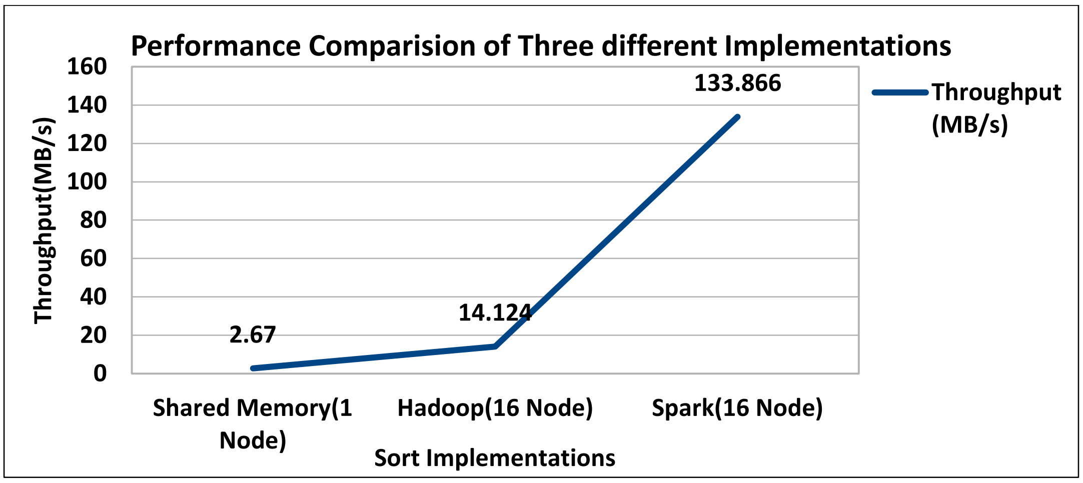


Conclusion:

- Shared Memory sort performance is less as compared to sort Map Reduce sort and Spark sort.
- Reason for being slow is In Shared Memory sort after completion of all chunks merging starts.
- However in Map Reduce and Spark parallel sorting and merging is performed by mapper and reducer.

Comparing Performance of Shared Memory on one node with 16 node Hadoop and Spark.

Performance of Three different Sort Implementations	
	Throughput(MB/s)
Shared Memory(1 Node)	**2.67**
Hadoop(16 Node)	**14.124**
Spark(16 Node)	**133.866**



Conclusion:

- Shared Memory sort performance is less as compared to 16 nodes Map Reduce sort and Spark sort.
- As no of scales are increased parallel execution increases thus giving higher throughput.
- Spark is much faster than Hadoop on 16 nodes.

➢ **Speedup graphs**

Speedup is a metric for improvement in performance between two systems processing the same problem.

It show the effect on performance after resource enhancement.

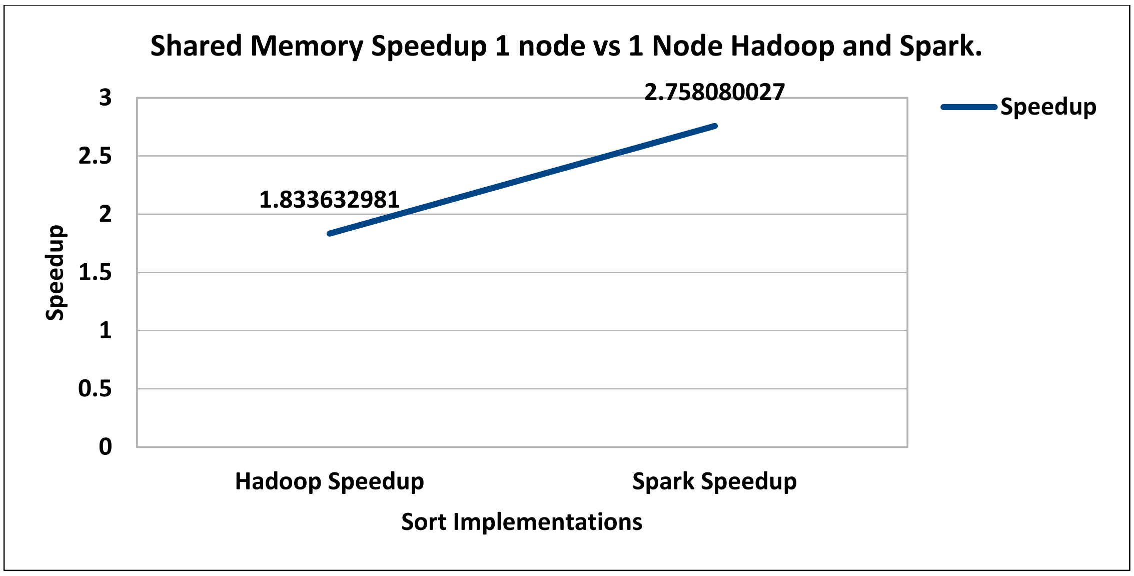
Speedup is calculated by as:

Speedup for shared memory for Hadoop = Throughput of Hadoop / Throughput of Shared Memory

Speedup for shared memory for Spark = Throughput of Spark/ Throughput of Shared Memory

Shared Memory Speedup 1 node vs 1 Node Hadoop and Spark.

Shared Memory Speedup 1 node vs 1 Node Hadoop and Spark		
	Hadoop Speedup	Spark Speedup
Speedup	1.833632981	2.758080027

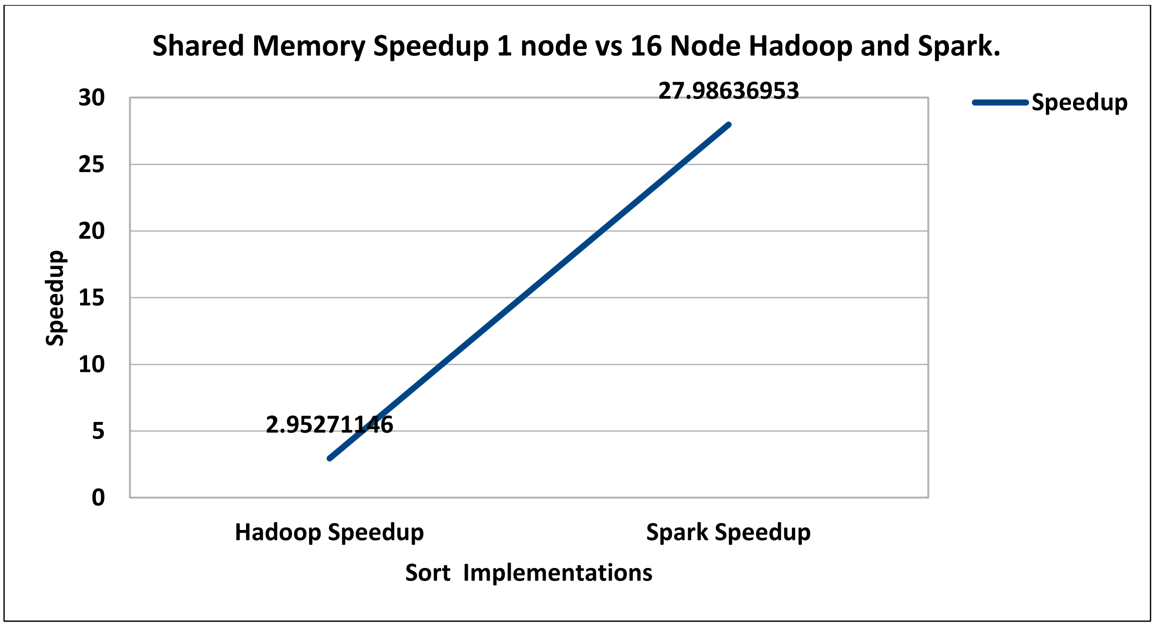


Conclusion:

- Spark speedup is greater than Hadoop speedup for 1 node.

Shared Memory Speedup 1 node vs 16 Node Hadoop and Spark.

Speedup Shared Memory 1 node vs 16 Node Hadoopa and Spark		
	Hadoop Speedup	Spark Speedup
Speedup	2.95271146	27.98636953

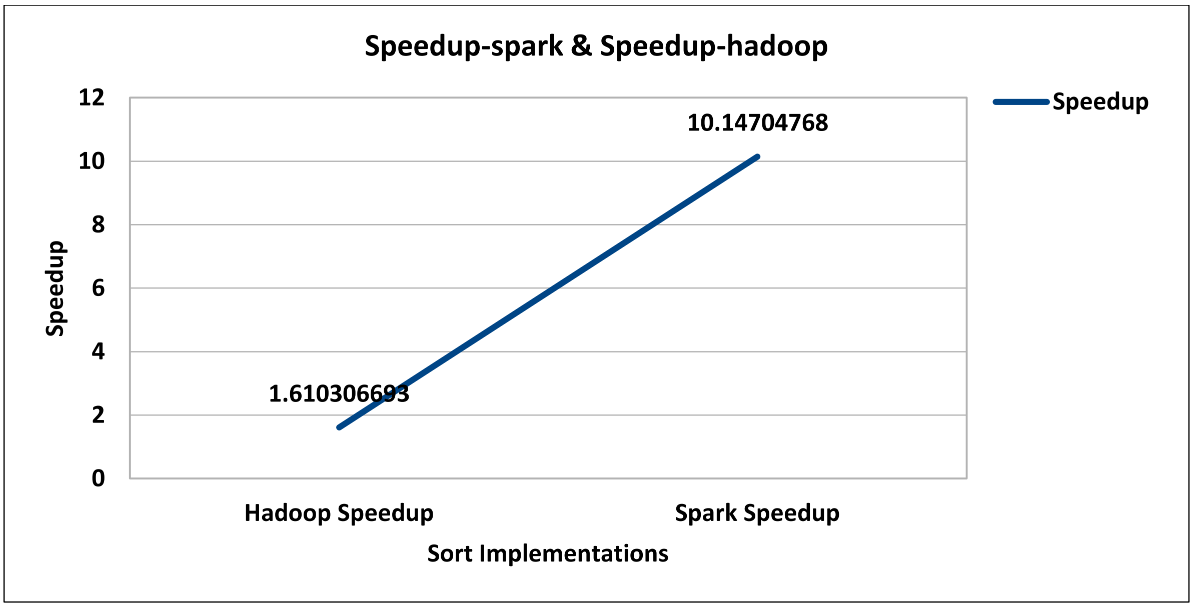


Conclusion:

- As scale increases Spark speedup increased by a great amount compared to Hadoop speedup for 16 node.

Speedup-Spark and Speedup-hadoop.

Speedup-spark & Speedup-hadoop		
	Hadoop Speedup	Spark Speedup
Speedup	1.610306693	10.14704768



Conclusion:

- From the above graph we can easily conclude that as number of nodes increases the speedup of Spark increases by a huge amount.
- Thus Spark gives better performance as number of nodes increases.

- **Which seems to be best at 1 node scale**
  - If we see the graph Spark gives a little better performance as compared to Hadoop.
  - It gives much better performance than shared memory.

- **Which seems to be best at 16 node scale.**
  - From above graph we can conclude that Spark system performs better on 16 node scale.

- **Which could be best at 100 node scale and 1000 node scale**
  - Looking to speedup graphs we can observe that Spark system speedup increases by a great amount as number of nodes increases.
  - Spark gives best result for resource enhancement.
  - Thus we can say that Spark will be best for 100 as well as 1000 nodes.

- **Compare with sort benchmark**
  - Looking towards the values of Baidu Sort and Triton sort we can say that Spark performs better than Hadoop.
  - Comparing their performance values to our experiment value we can relate that for their experiments also Spark gives better result when resources are added.
  - Spark is highly scalable and very much cost efficient than comparing to Hadoop.

- **What we learnt from Cloud sort benchmark?**
  - Cloud sort provides estimation for using public cloud infrastructure to sort large amount of files in cost effective way.
  - Cloud sort results can be used to sort fixed amount of entries of a file in a cost effective way.
  - Such benchmarks gives an efficient method for sorting large number of huge files.

- **Problems faced while setting up clusters**
  - Faced issue while creating cluster of 16 nodes Hadoop.
  - Not able to generate and add proper public keys and able to up dfs services.