CS553 CLOUD COMPUTING
HW6 Report

Team Members :

Yash Pradeep Gupte - A20472798

Amrutham Lakshmi Himaja - A20474105

In this assignment we have to write the Java code for performing sorting algorithms in different clusters of Spark and Hadoop using variant datasets and compare the outputs with MySort and Linux Sort.The goal is to observe the performance of Spark and Hadoop using 1 name node and multiple data nodes.

We have implemented 1small 1 large and 6small instances with 1gb, 4gb and 16g datasets.
The small instances are configured with 60GB disk space, 17GB ram and 4 cores.
The large instance is configured with 240GB disk space, 96GB ram and 24 cores.

HadoopSort

The input to the hadoop sort is the file generated by the gensort and the mapper class will produce the output of the key-value pairs in which the first 10 bytes will be the key and remaining is the value.The Partition function partitions the output of the map phase and sends it to reducers based on the first character of the key. The reducer is just an identity operation which writes input to the output file.

Table 1. Performance evaluation of sort (time in seconds)

Experiment	Shared Memory	Linux Sort	Hadoop Sort	Spark Sort
1 small.insatnce, 1GB dataset	41.670	23.510	56.260	84.205
1 small.insatnce, 4GB dataset	218.921	70.128	182.423	255.670
1 small.insatnce, 16GB dataset	285.773	121.526	346.240	595.700
1 large.instance, 1GB dataset	38.510	20.718	39.567	82.765
1 large.instance, 4GB dataset	207.992	63.67	154.500	220.113
1 large.instance, 16GB dataset	297.05	203.94	199.723	792.451
6 small.instances, 1GB dataset	N/A	N/A	33.244	72.011
6 small.instances, 4GB dataset	N/A	N/A	62.780	210.405

6 small.instances, 16GB dataset	N/A	N/A	355.012	805.720

Based on the experiments and table results ,for 1small instance and 1 large instance linux sort is faster and MySort is slightly slower but the results are better than hadoop and spark.

For 1GB and 4GB, Linux Sort and MySort all sort calculations in memory sort,which improves speed of computation,i/o and overall running time.For external memory sort, multithreading is implemented to run the code in parallel. Hadoop is faster on the small nodes than the large node because there are 4 mappers in Hadoop in a small instance and only one mapper in 1 large instance.

In the results hadoop running time is less than Spark therefore Hadoop gives better performances.

Factors that impact:

● Due to Storage limitations we were able to run 16gb instead of 64gb of data files.

● Spark uses a .coalesce() method for merging all the data partitions to one on disk.

Running hadoop and spark on 4 small instances as the data record size increases the performance of them is also better.Linux Sort and MySort perform better with smaller data

Hadoop and Spark are designed in a centralized manner, where the namenodes manages the datanodes. Parallelism is achieved as datanodes run simultaneously. For large datasets, there is large overhead of communication between namenode and datanodes thus it takes more time compared to Shared Memory implementation.

Q. How many threads, mappers, reducers, you used in each experiment?

> We have used 48 threads, 2 threads and 4 threads for the Shared memory and Linux sort implementations. For hadoop block size is 64MB , hence for 1gb data we get 16 mappers. For large instances we used 1 or 2 reducers and for small instances we used 4 to 8 reducers.

Q How many times did you have to read and write the dataset for each experiment?

For My Sort and Linux sort when the dataset fits in memory ,internal sorting is used and read and write will be done only once.When the dataset doesn't fit,external sorting is used and read and write will be done twice.

For hadoop, read and write will be done twice each .First read will start with the map and next read for reducer

Spark stores data on memory rather than the disk. If the dataset is larger than memory it will store on disk

Q What conclusions can you draw? Which seems to be best at 1 node scale (1 large.instance)? Is there a difference between 1 small.instance and 1 large.instance? How about 4 nodes (4 small.instance) ?

The 1 large instance is better than 1 small instance as there is more space and memory. The Mysort and Linux sort implementations were faster than hadoop and spark in small dataset cases because of the overhead of loading Java and Scala, starting clusters, and communication between nodes over a network.In case of large datasets Hadoop and Spark outperforms because they run in a distributed environment.

Q What speedup do you achieve with strong scaling between 1 to 6 nodes? What speedup do you achieve with weak scaling between 1 to 6 nodes?

For a 4gb dataset on scaling from 1small to 1 large nodes, hadoop performed better. Hadoop was faster on 1 large instance compared to 1 small.

For a 4gb dataset on 6 small nodes is better than 1 large node, thus weak scaling should improve the performance .

Q How many small.instances do you need with Hadoop to achieve the same level of performance as your shared memory sort?

To achieve the same level of performance as shared memory sort , Hadoop should have at least 2 times the current number of instances.

Q. Can you draw any conclusions on the performance of the bare-metal instance performance from HW5 compared to the performance of your sort on a large instance through virtualization?

Yes. The shared memory sort on bare metal was faster than the large instance. This is because of the virtualization cost which is a major factor.