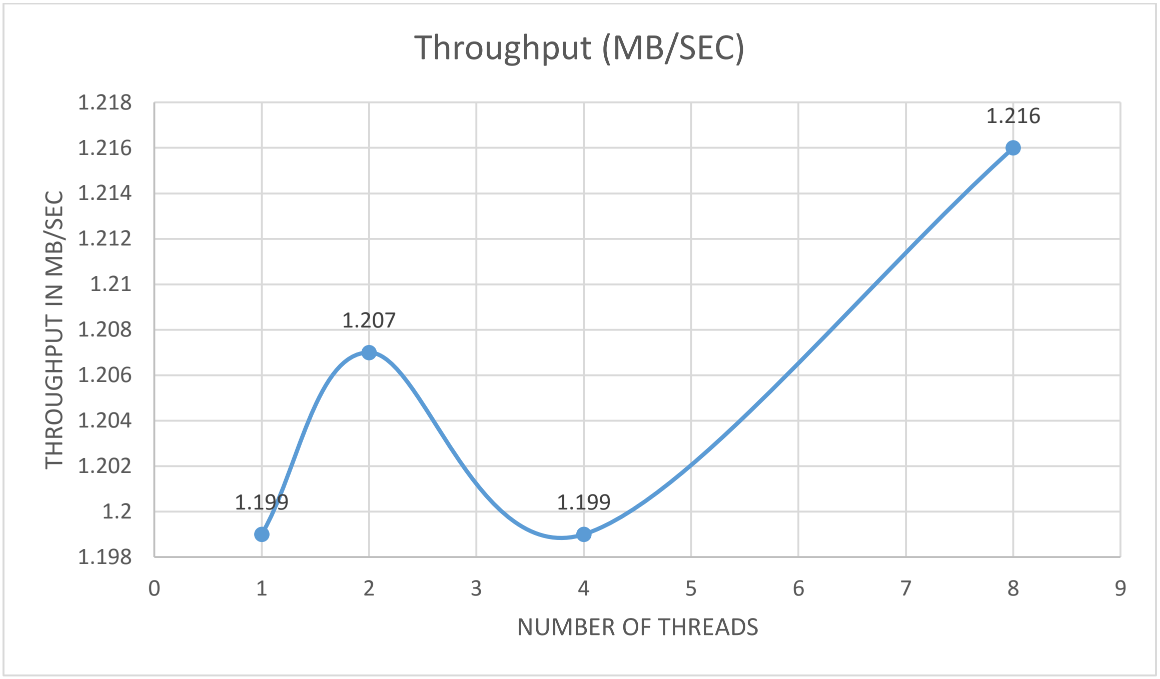
1) SHARED MEMORY SORT

10GB SORT – C3.Large

Sort time elapsed in minutes

1 thread	2 thread	4 thread	8 thread
8340secs	8280 secs	8340 secs	8220

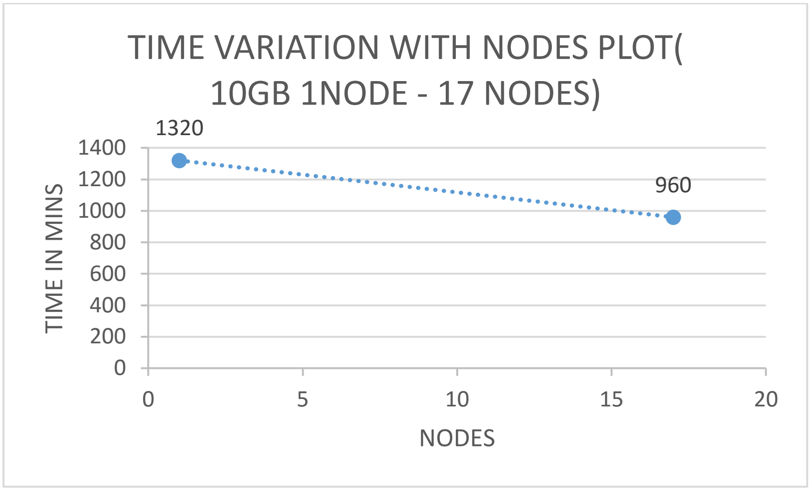
1 thread(MB/SEC)	2 thread(MB/Sec)	4 thread(MB/Sec)	8 thread(MB/SEC)
1.199	1.207	1.199	1.216



The maximum throughput is obtained in the case of 8 threads performing the shared memory sort. The file splitting time is constant in all the four scenarios and the difference is contributed by the variance in the sort time with multithreads.

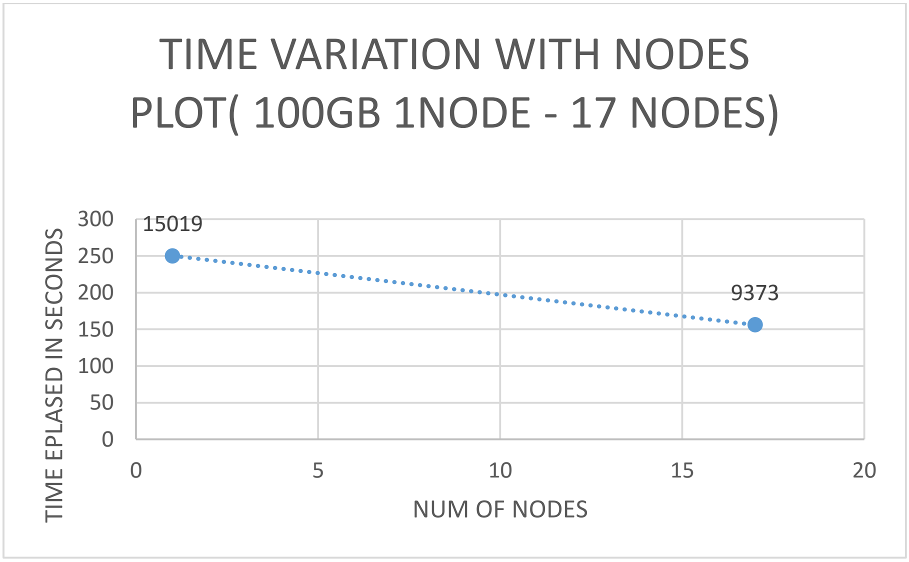
2) Hadoop

Size	Number of nodes	Time elapsed(SECONDS)	Throughput(MB/SEC)
10GB	1	1320	7.57575
10GB	17	960	10.41666



A) 100GB – 1 node - 4 hours 10mins 19 secs
100GB – 17 nodes – 2 hours 36 mins 13 sec

Size	Number of nodes	Time elapsed IN SECS	Throughput(MB/SEC)
100 GB	**1**	**15019**	**6.658**
100GB	**17**	**9373**	**10.668942**



From the above two charts it could be found that with the number of nodes increase there is a considerable increase in performance

HADOOP RESULTS

Size	**Number of nodes**	**Time elapsed IN SECS**	**Throughput(MB/SEC)**
10GB	**1**	**1320**	**7.57575**
100GB	**17**	**9373**	**10.668942**

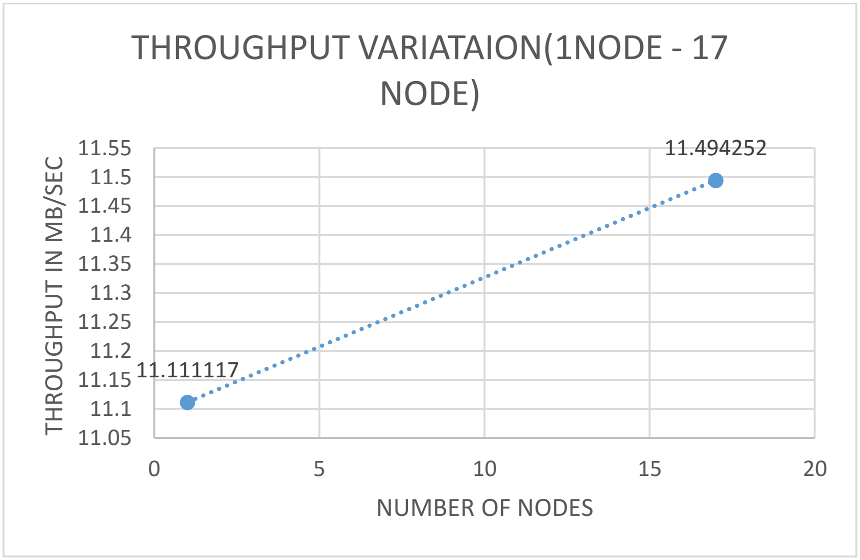
<u>SPEED UP CALCULATION HADOOP</u>

<u>Using the Throughput to calculate the baseline time and then speed up calculation.</u>

1) Speed up Hadoop = Time elapsed in single node /Time elapsed in 16 nodes
   10.668942/7.57575 = 1.408
2) Speed up Shared Memory = Time elapsed in shared memory sort / Time elapsed 16 nodes
   10.668/1.199= 8.8974

3) Spark

Size	Number of nodes	Time elapsed(SECONDS)	Throughput(MB/Sec)
10GB	1	900	11.11111 MB/SEC
100GB	17	8700	11.494252 MB/SEC



<u>Speed Up Calculation for Spark:</u>

<u>Since we are using different data sets in 1 node and 16 node set up , using the Throughput to calculate the baseline time and the speed up calculation . From throughput the</u>

1) Speed up spark = Time elapsed in single node /Time elapsed in 16 nodes
   11.494/11.111 = 1.03

2) Speed up Shared Memory = Time elapsed in shared memory sort / Time elapsed 16 nodes
11.494/1.199= 9.58

Speed up comparison Hadoop and Spark

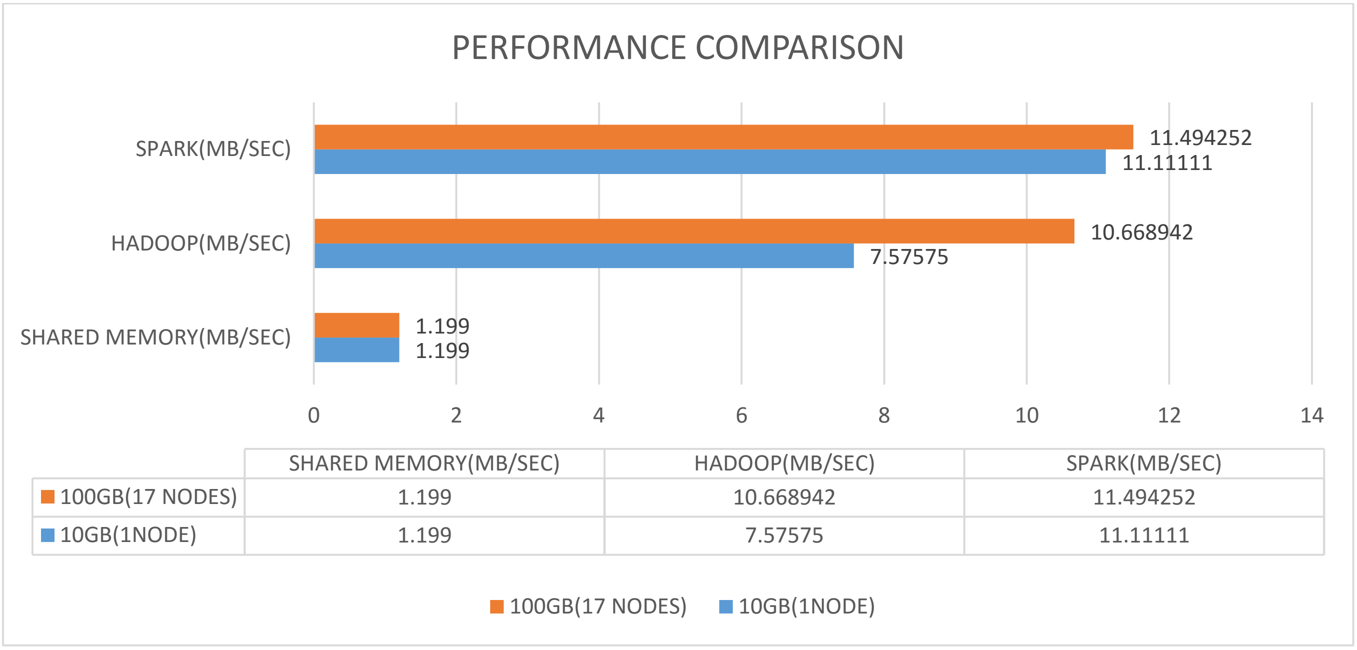
Speed up Hadoop = 1.404;

Speed up Spark = 1.03;

Speed up of Hadoop is better compared to the speed up of Spark

From the speed up shared memory values for Hadoop and spark, it could be inferred that the performance of Shared memory sort is weak compared to Hadoop and spark

3) **Shared Memory / Hadoop /Spark – Performance comparison**



	SHARED MEMORY(MB/SEC)	HADOOP(MB/SEC)	SPARK(MB/SEC)
🟧 100GB(17 NODES)	1.199	10.668942	11.494252
🟦 10GB(1NODE)	1.199	7.57575	11.11111

APACHE SPARK COMPARISON

Tera Sort Competition	Spark sort implementation
100 TB in 1,406 seconds	100 GB in 8700 seconds -- > 100TB should be done in approximately 8700000 seconds
207 Amazon EC2 i2.8xlarge nodes x	17 Amazon instances C3.large
32 vCores - 2.5Ghz , Intel Xeon E5-2670 v2	2 vCores - 2.8Ghz, Intel Xeon E5-2680 v2
244GB memory	3.75GB Memory
8x800 GB SSD	2 x 16 SSD

HADOOP COMPARISON

Hadoop Competition	Hadoop implementation
102.5 TB in 4,328 seconds	100 GB in 9373 seconds -- > 100 TB should be done in approximately 9373000 seconds
2100 nodes x	17 Amazon instances C3.large
2 2.3Ghz hexcore Xeon E5-2630	2 vCores - 2.8Ghz, Intel Xeon E5-2680 v2
64 GB memory,	3.75GB Memory
12x3TB disks	2 x 16 SSD

The number of threads, cores, available memory, hard drive capacity and most importantly an optimized code could help achieve better results. Decision on the number of nodes proportionately with the size of data to be sorted is important in performance improvement.

Cloud Sort Bench Mark

This paper tries to address the shortcomings of the existing sort benchmarks and proposes **a total cost of ownership benchmark**. For deciding on a particular cloud with various vendors available for public cloud, it becomes extremely important to have a benchmark on cost. External sorting becomes an ideal candidate for this as it is high IO intensive process which would cover network, storage, compute benchmarking. This would be a major step in deciding whether to go for a public cloud or a private cloud as well. This would definitely drive innovation on the public cloud sphere to provide best possible performance.

CONCLUSION

What conclusions can you draw? Which seems to be best at 1 node scale? How about 16 nodes? Can you predict which would be best at 100 node scale? How about 1000 node scales?

Data points are collected for Shared Memory, Spark, and Hadoop for 1 node and 16 node scenarios. With the throughput calculated in MB/sec for the three systems, it is found that **Spark has a better performance in both 1node and 16 node cases. As Spark performs better in 1 node and 16 node cases, it could be assumed for a 100 node scale as well Spark is going to perform better when compared to Shared Memory and Hadoop.**