Computer and Information Science; Vol. 8, No. 3; 2015
ISSN 1913-8989 E-ISSN 1913-8997
Published by Canadian Center of Science and Education

Hadoop Based Data Intensive Computation on IaaS Cloud Platforms

Sanjay P. Ahuja[1] & Sruthi Vijaykumar[1]

[1] School of Computing, University of North Florida, Jacksonville, Florida, USA

Correspondence: Sanjay P. Ahuja, School of Computing, University of North Florida, Jacksonville, Florida, USA. E-mail: sahuja@unf.edu

Received: July 13, 2015 Accepted: July 30, 2014 Online Published: July 31, 2015

doi:10.5539/cis.v8n3p103 URL: http://dx.doi.org/10.5539/cis.v8n3p103

This research was supported by the FIS Distinguished Professorship Award in Computer and Information Sciences awarded to Dr. Ahuja.

Abstract

Cloud computing is a relatively new form of computing, which uses virtualized resources and is dynamically scalable and is often provided as pay for use service over the Internet or Intranet or both. With increasing demand for data storage in the cloud, study of data-intensive applications is becoming a primary focus. Data intensive applications are those, which involve high CPU usage, processing large volumes of data typically in size of hundreds of gigabytes, terabytes or petabytes. This study was conducted on the Amazon's Elastic Cloud Compute (EC2) and Amazon Elastic Map Reduce (EMR) using HiBench Hadoop Benchmark suite.

HiBench is a Hadoop benchmark suite and is used for performing and evaluating Hadoop based data intensive computation on both these cloud platforms. Both quantitative and qualitative comparison was performed on both Amazon EC2 and Amazon EMR, including a study of their pricing models and measures are suggested for future studies and research.

Keywords: data intensive, cloud computing, hadoop, hibench, performance

1. Introduction

According to the National Institute of Standards and Technology (NIST), Cloud Computing can be defined as "A model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction" (Mell et al, 2011). There are three service models provided on the cloud.

1. Infrastructure-as-a-Service (IAAS) where the consumer is provided with the capability of provisioning storage, processing and networks and run arbitrary services. In this model, the consumer does not control the cloud infrastructure, storage and processing.
2. Platform-as-a-Service (PAAS) where the consumer is provided with the capability of deploying applications on the cloud using the provider's tools and, libraries and languages. In this model, the provider controls the infrastructure and the consumer only has access to deploy applications and change configuration settings related to deployment.
3. Software-as-a-Service (SAAS) where the consumer is provided with the capability of using provider's applications that are running on the cloud. In this case, the applications are either accessible from a web interface or a program interface. In this case also, the provider controls the cloud infrastructure.

1.1 Cloud Platforms

1.1.1 Amazon Elastic Cloud Compute (EC2)

Amazon EC2 also known as Amazon Elastic Compute Cloud is an IaaS cloud platform that provides a web service based API for provisioning, managing, and de-provisioning virtual servers inside the Amazon cloud. Applications residing anywhere on the Internet can launch a virtual server in the Amazon cloud and users can launch as many virtual servers as they want in the Amazon cloud. Amazon EC2 also allows users to configure security, and provide networking and scaling based on business requirements. Amazon EC2 instances can store data in Amazon S3 buckets or Amazon EBS (Elastic Block Storage). Amazon S3 provides an online file storage web service provided

by Amazon Web Service (Amazon, 2014).

Amazon EC2 instance types include:

On-Demand Instances where the user pays for computing capacity by the hour;

Reserved Instance (Light, Medium, and Heavy Utilization Reserved Instances) where the user pays one-time payment for the instance that they want to reserve and receive hourly discount on that instance;

Spot Instances where the users bid on unused EC2 instances and run the instances, as long the users bid does not exceed the spot price.

Each instance type varies in terms of memory capacity, available virtual cores, storage capacity and I/O performance. Users can chose the instance types based on their application needs.

1.1.2 Amazon Elastic Map Reduce (EMR)

Amazon EMR consists of multiple EC2 instances grouped in a cluster and can process huge amount of data by splitting the computational work across multiple EC2 instances and each EC2 instance is a virtual server. Amazon EMR cluster is managed by an open source Hadoop distribution (Noll, 2011). Amazon EMR cluster performance can be measured using Amazon CloudWatch. In order to run a job on Amazon EMR, users have to create an Amazon EMR job flow and execute it on the number of cluster nodes they need. Amazon EMR is suitable for large cloud computing as new instances can be easily configured (added and removed) on running custom code.

Amazon EC2 is a stand-alone instance whereas Amazon EMR is a cluster of EC2 instances. Cluster management is performed by the user on each Amazon EC2 instance whereas automated Cluster management occurs in Amazon EMR. They also differ with respect to the cost variance factor. Amazon EC2 is more cost effective than EMR since Amazon charges for cluster management .Amazon EMR pricing is the cost of running of Amazon EC2 instance plus the cost charged by Amazon for cluster management. Based on these varying factors, it is critical to establish benchmarks on both the clouds so that the user can determine whether to choose Amazon EC2 over Amazon EMR or vice-versa when it comes to Data Intensive Cloud Computing.

1.2 Data Intensive Computation

Data intensive applications are applications that involve high CPU usage, processing large volumes of data typically in size of hundreds of gigabytes, terabytes or petabytes. It has become critical that data intensive cloud providers provide on-demand computing instances and on-demand computing capacity. Clouds that provide on-demand computing instances and clouds that provide on-demand computing capacity like Amazon EC2 and Amazon EMR can support any computing model compatible with loosely coupled cluster. MapReduce along with Hadoop has become the dominant programming model used in data intensive cloud computing that provide on-demand computing capacity.

1.3 Hadoop

Apache Hadoop is an open source software project that enables distributed processing of large data sets across clusters of commodity servers (Hadoop, 2013). It utilizes master-slave system architecture (Hedger, 2011). Apache Hadoop is driven by two main components:

1. Map Reduce - The framework that understands and assigns work to the nodes in a cluster.

2. Hadoop Distributed File System (HDFS) - This file system spans all the nodes in a Hadoop cluster for data storage. It links together the file systems on many local nodes to make them into one big file system. HDFS assumes nodes will fail, so it achieves reliability by replicating data across multiple nodes.

1.4 MapReduce Programming Model

MapReduce is a programming model and software framework first developed by Google (MapReduce, 2014). This programming model helps in the processing of huge amount of data in parallel on large clusters in a reliable and a fault-tolerant manner. There are two fundamental steps associated with a MapReduce programming model. First step is the Map () function where a master node converts a set of data input into smaller set of data where individual elements are broken down into tuples (key-value pairs). Each of these tuples will be distributed to a slave node and these input list processed by the Map () function under slave nodes produces a different output list. The next step is the Reduce () function where the master node takes output provided by each of the worker

node and then combine them in a predefined way to provide the final output. MapReduce requires a "driver" method to initialize a job, which defines the locations of the input and output files and controls the MapReduce process. Each node in a MapReduce cluster is unaware of the other nodes in the cluster, and nodes do not communicate with each other except during the shuffling process.

2. Motivation and Related Work

Currently, there are no set of existing benchmarks and experiments for evaluating cloud performance of Amazon EC2 and Amazon EMR from the perspective of data intensive computing though there have been benchmarks that have been run on local machines and clusters using Hadoop. There also exist certain studies and benchmarking of Amazon cloud service particularly Amazon EC2 with other cloud platforms such as Rackspace as discussed below.

Huang et al in 'HiBench: A Representative and Comprehensive Hadoop Benchmark Suite' by Intel research group, talks about a comprehensive benchmark suite for Hadoop (Huang et al, 2010).. HiBench benchmarks according to the study can be divided into various categories: Data Benchmarks, Web search benchmarks and Analytical query benchmarks. This study on HiBench consists of a set of Hadoop programs including both synthetic micro-benchmarks and real-world applications.

Huang et al in 'The HiBench Benchmark Suite: Characterization of the MapReduce-Based Data Analysis', discuss the MapReduce model used as a prominent model for large-scale data analysis in the cloud. The authors use HiBench to evaluate and characterize Hadoop framework in terms of speed (job running time), throughput (the number of tasks completed per minute), HDFS bandwidth, system resource (CPU, memory and I/O) utilizations, and data access patterns such as map period , average mapper time and job execution time. The authors concluded that HiBench is a new, realistic and comprehensive benchmark suite for Hadoop, which consists of a set of Hadoop programs including both synthetic micro-benchmarks and real-world applications. The HiBench suite is essential for the community to properly evaluate and characterize Hadoop, because it's workload not only represent a wide range of large-scale data analysis using Hadoop, but also exhibit very diverse behaviors in terms of data access patterns and resource utilizations.

According to the recent benchmark study on clouds by Sarda et al in 'Cloud Performance Benchmark – Amazon EC2 vs. RackSpace' (cloud based VPS), Rackspace's 512MB instance was more than twice as fast as Amazon's micro instance (Sarda et al, 2011). The study benchmarked metrics Relative CPU Performance, IO Read and IO Write, Number of Requests Apache Can Handle and Processing Power. The authors concluded that Rackspace is 3 times faster than Amazon EC2 in terms of Processing Power, Rackspace can handle 5.5 times more requests than Amazon when using Apache HTTP server, and Rackspace can write 7.6 times more data than Amazon per second and is 2.3 times faster than Amazon EC2 in terms of CPU performance.

As discussed above, there are various benchmarks comparing the performance of Amazon EC2 to other clouds and vice versa but there do not exist any benchmarks studies that focus on comparing the performance of Amazon EC2 versus Amazon EMR for data intensive computing using Hadoop which is the present experimentation.

3. Methodology

3.1 Hardware Configuration

Hardware configuration used is very critical for Hadoop based data intensive benchmark. For this purpose, an M3 General Purpose Double Extra Large instance type was chosen for both Amazon EC2 and Amazon EMR. Amazon M3 instance types provide a balance of memory, compute and network resources with its most prominent features being SSD based storage for very fast I/O performance and High Frequency Intel Xeon E5-2670 v2 (Ivy Bridge) Processors.

Table 1. Hardware Configuration

Component	Specification
Instance Type	M3.2xlarge
Processor	Intel Xeon E5-2670 v2 2.5 GHz
Memory	30GB
Storage Drives	160 GB (2 * 80 GB SSD)
I/O Performance Adaptor	High / 1000 Mbps

3.2 Software Configuration

Install Amazon Linux AMI on the workstations. Install version 1.7 of the Java JDK. Install Hadoop version 1.0.3 on both Amazon EC2 and Amazon EMR. Install Hi-Bench 2.2 on Amazon EC2 and Amazon EMR Hadoop. Configure SSH on all the nodes on Amazon EC2 and Amazon EMR for communication between name node and all the data nodes. Install Python on Amazon EC2, which is a pre-requisite for Star Cluster Installation. Use StarCluster open source toolkit to create cluster on Amazon EC2.Create cluster on Amazon EMR using Amazon UI (StarCluster, 2011).

3.3 Benchmarks

Table 2. Benchmarks and Metrics

Benchmarks	Method	Metrics Measured
Micro Benchmarks	Sort	Response Time
	WordCount	Data Size
	TeraSort	Throughput
Web Search	Page Ranking	Response Time
		Page Workload
		Throughput
Analytical Query	Hive Join	Execution Time
	Hive Aggregation	Data Size
		Throughput

HiBench is a representative and comprehensive benchmark suite for Hadoop. This benchmark suite consists of a set of Hadoop programs including both synthetic micro-benchmarks and real-world applications. These benchmarks are used intensively for Hadoop benchmarking, tuning and optimizations. The categories of benchmarks used for this research are Micro benchmarks (Sort, WordCount, and TeraSort) which include more of unstructured data; Web Search benchmarks (PageRank) which includes more of semi-structured data and Analytical Query benchmarks (Hive Join, Hive Aggregation) which includes structured data (Wang, 2014).

3.3.1 Micro Benchmarks

1. **Sort:** This workload sorts its text input data, which is generated using the Hadoop RandomTextWriter program. Here the sorting is done automatically during the Shuffle and Merge stage of MapReduce programming model. This is an I/O bound function. The input workload for the Sort benchmark is datasize to be generated.
2. **WordCount:** This workload counts the occurrence of each word in the input data, which are generated using the Hadoop RandomTextWriter program. This job extracts a small amount of information from a large data source hence this is a CPU bound function. The input workload for the WordCount benchmark is the datasize to be generated.
3. **TeraSort:** This is a benchmark where input data is generated by Hadoop TeraGen program that creates by default 1 billion of 100 bytes lines. The data here are then sorted by Terasort which provides its own input and output format and also its own Partitioner which ensures that the keys are equally distributed among all nodes. This is an improved Sort program, which provides equal loads between all the nodes during the test. As a result, this is a CPU bound function for the Map stage and I/O bound function for the Reduce stage. The input workload for the Terasort benchmark is datasize to be generated.

3.3.2 Web Search Benchmark

PageRank: The workload contains an implementation of the PageRank algorithm on Hadoop, which is a link analysis algorithm, used widely in web search engines. This is a CPU bound function. The input workload to PageRank algorithm is number of Wikipedia pages.

3.3.3 Analytical Query Benchmark

Hive Join and Hive Aggregation: The workload contains queries that correspond to the usage profile of business analysts and other database users. The two tables created are User Rankings table and UserVisits table. Once the data source has been generated, two of the Hive requests would be performed, a Join and an Aggregation. These tests are I/O bound functions. The input workload for the Analytical Query Benchmark is

number of records to be inserted into User Rankings table and User Visits table. The overview of benchmarks, their categories and metrics captured are shown in Table 2.

4. Results and Discussion

The study evaluates and compares the performance of the Amazon EC2 and Amazon EMR cloud services using HiBench benchmark suite, which includes Micro Benchmarks (Sort, WordCount, Terasort), Web Search benchmark (Page Rank) and Analytical Query performance Benchmarks (Hive Join and Hive Aggregation). Microsoft Excel 2010 built in function T-TEST was used for statistical analysis of the results obtained from the benchmarks on Amazon EC2 and Amazon EMR. The T-TEST function used two datasets as input, first dataset being Amazon EC2 and second dataset being Amazon EMR. The p-value was computed; a p-value of exceeding 0.05 is considered statistically insignificant difference between the two datasets, while a p-value not exceeding 0.05 an indication of statistically significant difference between the two datasets (Tables 4 – 15).

For each benchmark, the response time (in seconds) and throughput (in megabytes per sec) is measured with increasing number of nodes from 1 to 8. Graphs were then plotted for Amazon EC2 and Amazon EMR cloud services for comparing their performance. The graphs compare the performance of Amazon EC2 and Amazon EMR cloud services using each of the HiBench benchmark suite, which includes Micro Benchmarks (Sort, WordCount, Terasort), Web Search benchmark (Page Rank) and Analytical Query performance Benchmarks (Hive Join and Hive Aggregation) by varying the dataset size (1GB, 10GB, 100GB) to represent data intensive computation using Hadoop. For each graph, the y-axis represents the response time/throughput values achieved during the tests, and the x-axis represents the number of nodes tested (Figures 1 – 18).

Table 3 provides a basic insight into the pricing of Amazon EMR and Amazon EC2 for an m3.2xlarge instance. Amazon EC2 has a base price of $0.560/hr per instance whereas Amazon EMR pricing is cost of an Amazon EC2 instance which is $0.560/hr per instance plus the cost that Amazon charges for cluster management for Amazon EMR which is $0.140 /hr totaling $0.700 /hr. As the number of nodes are increased, the variation becomes more significant as shown in Figure 24 below. The variation becomes drastically significant when the number of nodes are multiplied by number of hours times price per instance.

Table 3. Cloud Pricing

Nodes	Amazon EC2 (per hour)	Amazon EMR(per hour)
	m3.2xlarge instance	m3.2xlarge instance
1	$0.56	$0.70
2	$1.12	$1.40
3	$1.68	$2.10
4	$2.24	$2.80
5	$2.80	$3.50
6	$3.36	$4.20
7	$3.92	$4.90
8	$4.48	$5.60

5. Conclusions

The Amazon EC2 and Amazon EMR cloud services were tested using the HiBench benchmark suite while the number of nodes (1 to 8) and the size of the dataset (1GB, 10GB, and 100GB) were varied. Overall, it appeared that Amazon EC2 was well suited for less data intensive applications for data size less than 100 GB. The results of datasets of 1GB and 10GB run on m3.2xlarge instance showed this behavior. When we move over to higher benchmark workloads of 100 GB, Amazon EMR preformed better than Amazon EC2. This can be attributed to the fact that Amazon EMR installation of Hadoop containing patches and improvements added to Apache Hadoop to make it work effectively on AWS. This also includes using better compression codec's and fixes to better combine and split input files and better performance tuning of running clusters on Amazon EMR. The configuration settings of Hadoop used for Amazon EMR cluster are optimized for scalability and more data intensive applications which explain why Amazon EMR performed better than Amazon EC2 on larger data sets.

For Sort, TeraSort, Page Rank and Hive Aggregate benchmarks, the difference in response time between Amazon EC2 and Amazon EMR and the difference in throughput between Amazon EC2 and Amazon EMR was more significant than in WordCount and Hive Join benchmarks as the former contains more data intensive and I/O operations compared to the latter.

Certain advantages that Amazon EMR has over Amazon EC2 is that the Amazon EMR can be used for large scale data processing that includes a lot of setting and configuration work as Amazon steps forward to remove that extra work out for the customers. Also Amazon takes care of cluster monitoring, resource management, cluster start-up and shutdown and even security groups management in case of Amazon EMR. Most of the cases it is hard to tune the performance of running clusters but in case of Amazon EMR, it takes care of performance tuning of the clusters while running a job or a workload. Even Hadoop is made simple and easy by Amazon EMR. Certain benefits of EMR are:

1. Elastic: Amazon EMR uses many in few EC2 instances as needed. Also spins large or small job flows in minutes.

2. Easy to use: Easy to run jobs quickly using the web console. No detailed configuration is required.

3. Reliable: Fault tolerant service build on top of the Amazon Web Service (AWS) infrastructure.

4. Cost Effective: Amazon monitors the progress of each job flow and turn off the resources when job flow is done.

From a scaling and cost perspective, for higher workloads and large number of nodes to be managed, it is better to opt for Amazon EMR than Amazon EC2 even though the cost of Amazon EMR is higher than that of EC2. Amazon EMR automatically takes care of performance tuning of running clusters, cluster monitoring, resource management and security groups management. It is also fault tolerant and automatically retires failed tasks where as in Amazon EC2, all these will have to be done manually. There is less overhead in Amazon EMR compared to Amazon EC2 where as in case of small datasets and applications that don't need much scalability and need to operate on low cost , Amazon EC2 is an better option.

6. Future Work

This study is limited to benchmarking the two cloud services provided by Amazon, Amazon EC2 and Amazon EMR cloud services to evaluate the performance of Hadoop on data intensive applications while varying workloads and the number of nodes in the cluster. Extensions to this study on cloud performance include evaluating the performance of these cloud service on Bigdata level that is, varying the sizes upto terabytes of data. This may help the research to evaluate the performance pattern of Hadoop on each node for both the cloud services thus helping in further analysis.

In this study, we utilized m3.2xlarge instance provided by Amazon, which provides a balance of compute, memory and network resources. So further studies could be conducted on various instance types provided by Amazon, such as compute optimized instances (C3 instances), storage optimized instances (I2 instances) and Graphic optimized instances (G2 instances), to further explore the benchmarking on Amazon cloud platform. Another scope of further research is in terms of new benchmarks to that could be used for evaluating the performance. The research utilizes HiBench benchmark suite, which is a set of Hadoop benchmarks. New benchmarks can be introduced to experiment with Hadoop performance on data intensive applications.

References

Amazon. (2014). What is Amazon EMR. Retrieved November 20, 2014, from http://docs.aws.amazon.com/ElasticMapReduce/latest/DeveloperGuide/emr-what-is-emr.

Hadoop, (2013). *HDFS Architecture Guide*. Retrieved November 20, 2014, from http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

Hedger, D. A. (2011). *Hadoop Design, Architecture & MapReduce Performance*, DHT Technologies, 2011.

Huang, S., Huang, J., Dai, J., Xie, T., & Huang, (2010). *The HiBench Benchmark Suite: Characterization of the MapReduce-Based Data Analysis*, Conference: Data Engineering Workshops (ICDEW), Intel China Software Center, Shanghai, China, 2010.

Huang, S., Huang, J., Liu, Y., Yi, L., & Dai, J. (2010). *HiBench: A Representative and Comprehensive Hadoop Benchmark Suite.* Intel Asia-Pacific Research and Development Ltd., Shanghai, P.R.China, 2010.

MapReduce. (2014). MapReduce: Overview. Retrieved November 20, 2014, from http://gppd-wiki.inf.ufrgs.br/index.php/MapReduce

Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing*. Recommendations of the National Institute of Standards and Technology, Special Publication 800-145, September 2011.

Noll, M. G. (2011). *Running Hadoop on Ubuntu Linux (Multi-Node Cluster)*. Retrieved November 20, 2014, from http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-multi-node-cluster/

Sarda, K., Sanghrajka, S., & Sion, R. (2011). *Cloud Performance Benchmark -Amazon EC2 vs. Rackspace*, Cloud Commons Online, 2011.

StarCluster. (2014). Installing StarCluster, Retrieved November 20, 2014, from http://star.mit.edu/cluster/docs/0.93.3/installation.html

Wang, (2014). *Hadoop Benchmark Suite (HiBench)*. Retrieved November 20, 2014, from https://github.com/intel-hadoop/HiBench/

Appendix A

Table 4. Sort: Response Time - Amazon EC2 vs. Amazon EMR

SORT						
Data size	**1GB**		**10GB**		**100GB**	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	133.411	189.75	385.191	535.754	4586.868	3157
2	74.301	116.994	210.991	303.568	2322.305	1462.162
3	53.251	95.31	153.914	240.459	1691.882	935.593
4	42.286	81.812	126.921	179.409	1194.463	749.433
5	41.282	75.251	119.811	171.431	1083.334	731.268
6	32.28	70.913	107.146	140.411	809.1	555.417
7	31.306	65.884	102.942	134.376	785.187	585.735
8	30.238	64.919	101.08	122.371	728.133	583.582
P-value	**1.07483E-06**		**0.00367057**		**0.008840098**	

Table 5. Sort- Throughput – Amazon EC2 vs. Amazon EMR

SORT						
Data size	**1GB**		**10GB**		**100GB**	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	1.964130	1.380896	6.801263	4.889903	5.711348	9.306178
2	3.526689	2.239645	12.416574	8.629978	11.280689	17.916840
3	4.920781	2.749187	17.021098	10.894934	15.484059	28.000768
4	6.196767	3.202770	20.641071	14.602305	21.932199	34.956190
5	6.347476	3.482014	21.865984	15.281863	24.182016	35.824516
6	7.163937	3.695021	24.450613	18.657976	32.378199	44.725577
7	7.702596	3.977066	25.449140	19.495930	33.364282	45.844257
8	8.388389	4.036184	26.124556	21.408545	35.978593	48.120618
P-value	**0.000551**		**0.000038**		**0.000058**	

Table 6. WordCount: Response Time – Amazon EC2 vs. Amazon EMR

WORD COUNT						
Data size	**1GB**		**10GB**		**100GB**	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	131.466	216.949	416.625	465.245	2880	2779.16
2	73.357	124.694	247.451	275.105	1492.06	1445.108
3	53.338	93.914	202.412	236.128	1029.257	968.248

4	42.366	79.011	165.394	188.335	807.6	760.07
5	41.338	69.013	133.313	166.888	674.592	626.921
6	32.361	66.028	127.368	159.003	554.232	504.806
7	31.406	63.918	113.332	143.915	517.415	463.774
8	30.357	60.062	95.313	128.991	453.212	428.989
P-value	**0.000409808**		**4.70076E-06**		**0.000203863**	

Table 7. WordCount: Throughput – Amazon EC2 vs. Amazon EMR

WORD COUNT						
Data size	**1GB**		**10GB**		**100GB**	
#nodes	**EC2**	**EMR**	**EC2**	**EMR**	**EC2**	**EMR**
1	1.993297	1.207790	6.288044	5.630873	9.013599	9.426330
2	3.572267	2.096835	10.586970	9.522675	17.558458	18.128249
3	4.913023	2.790095	12.942692	11.094557	25.452656	27.056372
4	6.185404	3.316360	15.839488	13.909978	32.438490	34.466928
5	6.339223	3.796806	19.651169	15.697567	38.834324	41.787208
6	7.144061	3.795980	20.568402	16.454793	47.260035	51.895735
7	8.086490	4.099455	23.115768	18.203354	50.631165	56.487165
8	8.357011	4.362641	27.485823	20.309445	57.803686	61.059823
P-value	**0.000338**		**0.004696**		**0.005539**	

Table 8. TeraSort: Response Time – Amazon EC2 vs. Amazon EMR

TERASORT						
Data size	**1GB**		**10GB**		**100GB**	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	141.919	217.436	435.153	568.197	4772.874	4077.234
2	79.861	130.358	233.105	300.937	3706.249	2034.876
3	57.802	98.117	206.128	219.835	2041.644	1192.721
4	50.757	81.5	188.335	193.827	1501.415	1082.909
5	48.759	81.795	174.888	185.84	1397.962	1054.86
6	37.786	73.79	169.003	175.776	1151.253	949.861
7	34.767	66.426	143.915	152.785	1137.792	928.615
8	33.829	65.798	128.991	139.794	1110.719	917.558
P-value	**0.000125555**		**0.043215098**		**0.01498451**	

Table 9. TeraSort: Throughput – Amazon EC2 vs. Amazon EMR

TERASORT						
Data size	**1GB**		**10GB**		**100GB**	
#nodes	**EC2**	**EMR**	**EC2**	**EMR**	**EC2**	**EMR**
1	7.215379	4.709429	23.531941	18.021909	38.841957	46.974741
2	12.822288	7.855296	43.852495	34.027042	42.887788	51.894589
3	17.715663	10.436527	55.826915	46.580372	50.155634	57.781176
4	20.174572	12.564426	61.498892	56.628690	68.202295	85.023325
5	21.001266	12.519112	63.098644	61.746238	73.249451	91.293951
6	27.100004	13.877229	66.755736	63.569012	79.409778	97.074445
7	29.453238	15.415662	69.441592	67.410955	89.045156	101.212000
8	30.269909	15.562794	70.872872	68.360522	92.192489	111.600519
P-value	**0.000684**		**0.003932**		**0.000104**	

Table 10. PageRank: Response Time – Amazon EC2 vs. Amazon EMR

PAGE RANK

Data size	PAGES=100000		PAGES=1000000		PAGES=10000000	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	237.249	414.082	428.006	590.163	2635.857	2011.837
2	136.084	239.756	229.716	305.56	1385.46	1069.471
3	102.07	185.172	169.714	236.515	972.505	765.41
4	84.991	154.173	134.643	191.506	748.718	586.472
5	84.03	138.526	124.639	173.498	665.708	533.651
6	65.016	127.154	102.628	155.443	539.797	433.785
7	62.005	120.546	93.662	146.74	504.606	414.661
8	60.971	111.72	89.625	132.542	482.406	382.664
P-value	**0.000855784**		**0.001379679**		**0.011394124**	

Table 11. PageRank: Throughput – Amazon EC2 vs. Amazon EMR

PAGE RANK						
Data size	**PAGES=100000**		**PAGES=1000000**		**PAGES=10000000**	
#nodes	**EC2**	**EMR**	**EC2**	**EMR**	**EC2**	**EMR**
1	0.067532	0.038692	0.433854	0.352804	0.795236	1.041898
2	0.117735	0.066826	0.808356	0.607711	1.512948	1.959968
3	0.156969	0.086524	1.094148	0.785118	2.155391	2.738570
4	0.188512	0.103921	1.379146	0.969642	2.799624	3.574133
5	0.190668	0.115659	1.489841	1.070285	3.148721	3.927902
6	0.246429	0.126003	1.809373	1.194601	3.883179	4.723298
7	0.258396	0.132911	1.982579	1.265451	4.153991	5.055042
8	0.262778	0.143411	2.071881	1.401007	4.345155	5.478013
P-value	**0.000260**		**0.001116**		**0.000169**	

Table 12. Hive Join: Response Time – Amazon EC2 vs. Amazon EMR

HIVE JOIN						
Data size	**USERVISITS=1000000 PAGES=600000**		**USERVISITS=10000000 PAGES=6000000**		**USERVISITS=100000000 PAGES=60000000**	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	212.977	438.379	325.872	496.786	1005.85	988.224
2	139.555	296.381	232.214	342.805	601.309	596.259
3	113.582	254.365	206.048	298.041	492.683	465.918
4	100.074	229.463	191.099	271.871	407.333	392.035
5	97.372	218.134	185.204	256.597	398.057	354.6
6	86.203	207.355	176.935	246.723	366.065	334.557
7	84.105	204.527	169.979	237.474	352.066	321.241
8	82.084	195.978	168.082	226.963	324.746	302.633
P-value	**1.2477E-05**		**0.00021051**		**0.000669366**	

Table 13. Hive Join: Throughput – Amazon EC2 vs. Amazon EMR

HIVE JOIN						
Data size	USERVISITS=1000000 PAGES=600000		USERVISITS=10000000 PAGES=6000000		USERVISITS=100000000 PAGES=60000000	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	0.391939	0.190415	2.560921	1.679863	8.296048	8.444017
2	0.598145	0.281645	3.593808	2.434424	13.877358	13.994892
3	0.734924	0.328167	4.050185	2.800060	16.937017	17.909976
4	0.834124	0.363780	4.367017	3.069590	20.485893	21.285294
5	0.831647	0.382673	4.506018	3.252308	20.963280	23.532375
6	0.968343	0.402566	4.716606	3.382468	22.795351	24.942178
7	0.973575	0.408132	4.909622	3.514206	23.701750	25.976075
8	1.000152	0.425936	5.068703	3.676954	25.695714	27.573266

P-value	0.000069	0.004796	0.005515

Table 14. Hive Aggregation: ResponseTime – Amazon EC2 vs. Amazon EMR

HIVE AGGREGATION						
Data size	**USERVISITS=1000000 PAGES=600000**		**USERVISITS=10000000 PAGES=6000000**		**USERVISITS=100000000 PAGES=60000000**	
#nodes	EC2	EMR	EC2	EMR	EC2	EMR
1	114.241	190.076	182.326	222.487	757.439	629.469
2	74.833	121.193	122.613	150.266	410.428	341.373
3	59.709	97.942	106.445	124.895	312.87	255.964
4	53.711	88.506	97.445	110.119	254.585	216.602
5	52.706	81.922	96.396	109.438	230.27	199.396
6	45.607	75.678	89.424	102.394	192.116	165.451
7	45.6	73.822	87.334	98.926	182.022	162.224
8	45.57	68.617	85.418	96.013	179.141	159.215
P-value	**0.000352441**		**0.001563433**		**0.007024392**	

Table 15. Hive Aggregation: Throughput – Amazon EC2 vs. Amazon EMR

HIVE AGGREGATION						
Data size	**USERVISITS=1000000 PAGES=600000**		**USERVISITS=10000000 PAGES=6000000**		**USERVISITS=100000000 PAGES=60000000**	
#nodes	**EC2**	**EMR**	**EC2**	**EMR**	**EC2**	**EMR**
1	0.513319	0.308519	3.215524	2.635092	7.739821	9.313314
2	0.783640	0.483874	4.781497	3.901572	14.283730	17.173129
3	0.982132	0.598743	5.507762	4.694133	18.737631	22.903387
4	1.091808	0.662578	6.016457	5.324001	23.027447	27.065506
5	1.112627	0.715829	6.081930	5.357131	25.458994	29.401004
6	1.285814	0.774890	6.556111	5.725664	30.515119	35.433105
7	1.286011	0.794372	6.713006	5.926386	30.530057	35.268328
8	1.296395	0.854630	6.776122	6.106191	32.725298	36.820919
P-value	**0.000011**		**1.19547E-07**		**0.000023**	

Appendix B

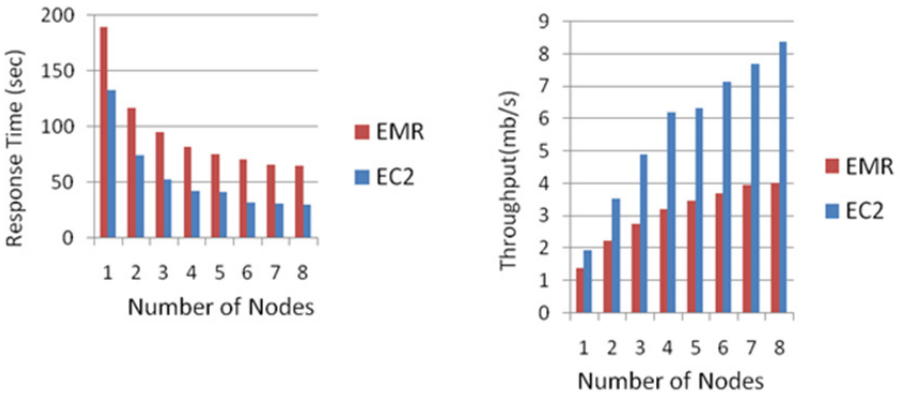

Figure 1. Sort – Amazon EC2 vs Amazon EMR (1 GB)

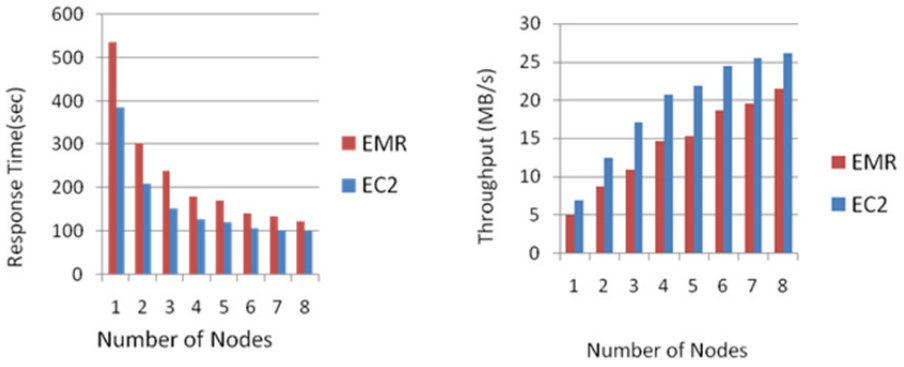

Figure 2. Sort – Amazon EC2 vs Amazon EMR (10 GB)

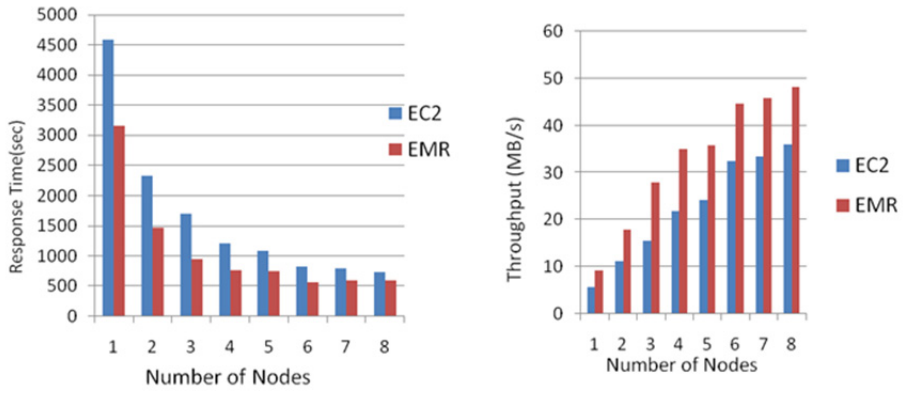

Figure 3. Sort – Amazon EC2 vs Amazon EMR (100 GB)

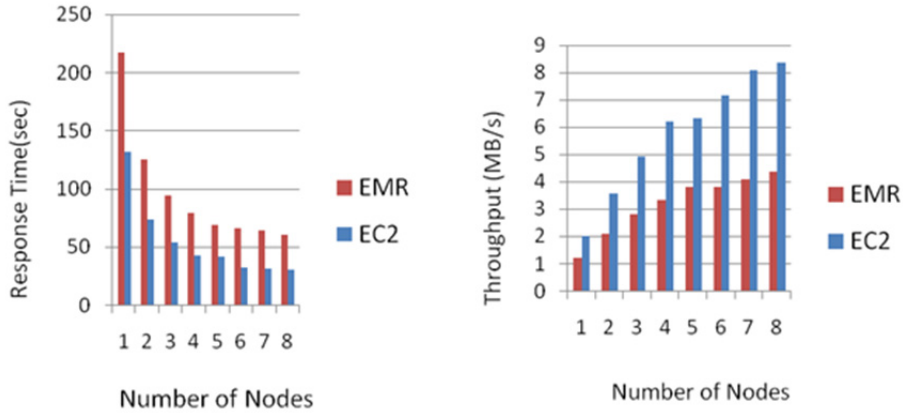

Figure 4. WordCount – Amazon EC2 vs Amazon EMR (1 GB)

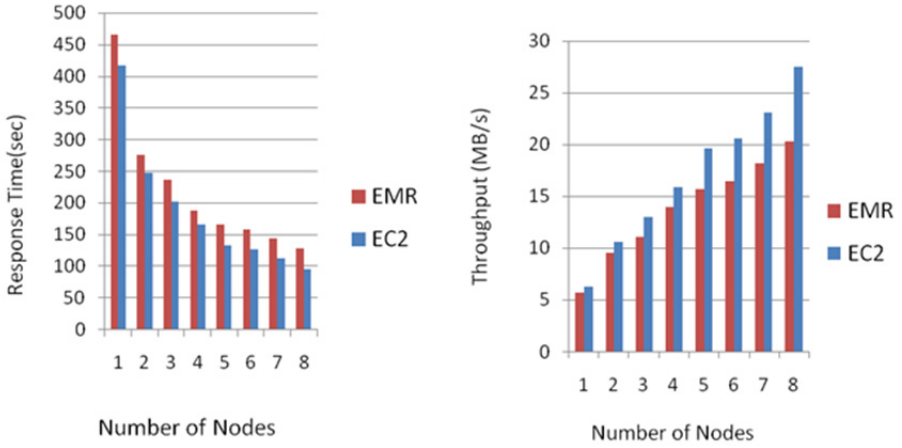

Figure 5. WordCount – Amazon EC2 vs Amazon EMR (10 GB)

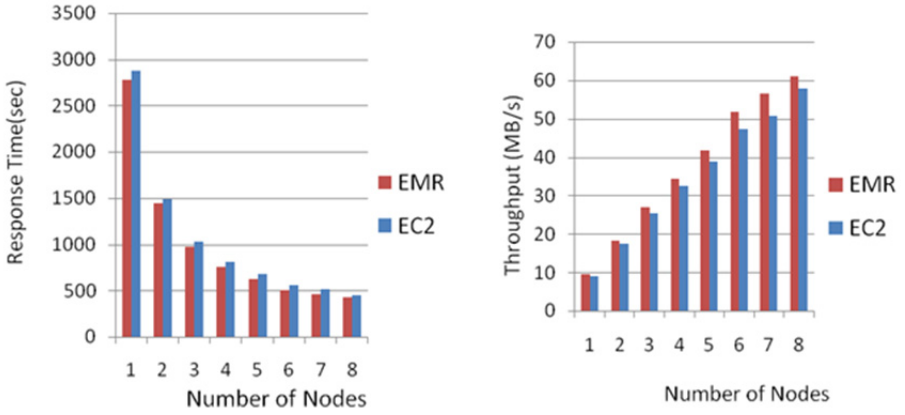

Figure 6. WordCount – Amazon EC2 vs Amazon EMR (100 GB)

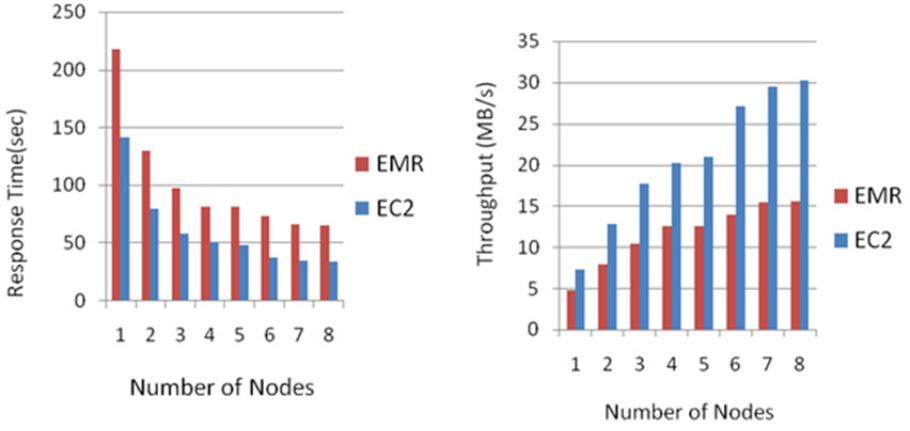

Figure 7. TeraSort – Amazon EC2 vs Amazon EMR (1 GB)

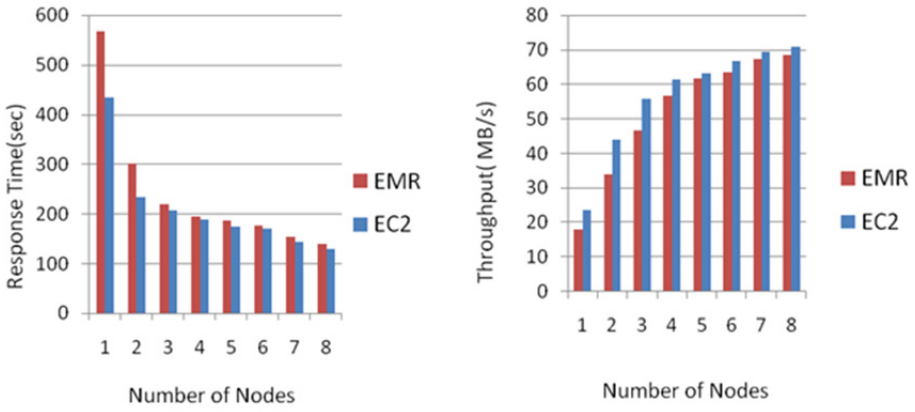

Figure 8. TeraSort – Amazon EC2 vs Amazon EMR (10 GB)

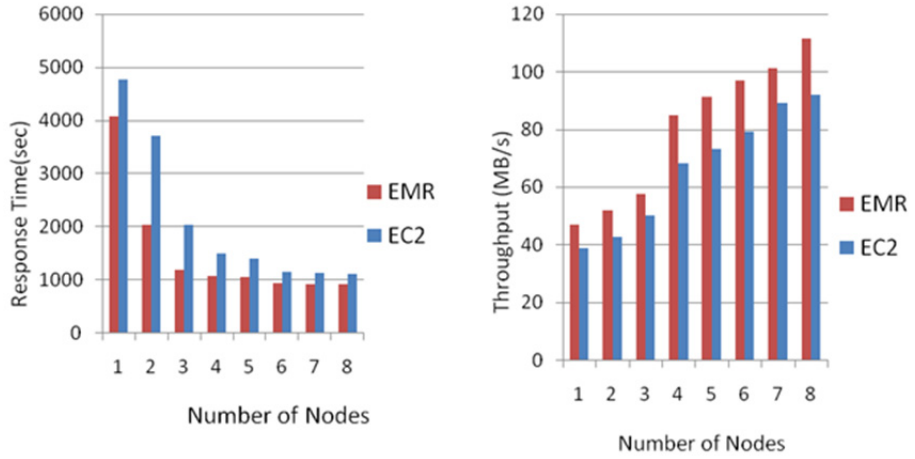

Figure 9. TeraSort – Amazon EC2 vs Amazon EMR (100 GB)

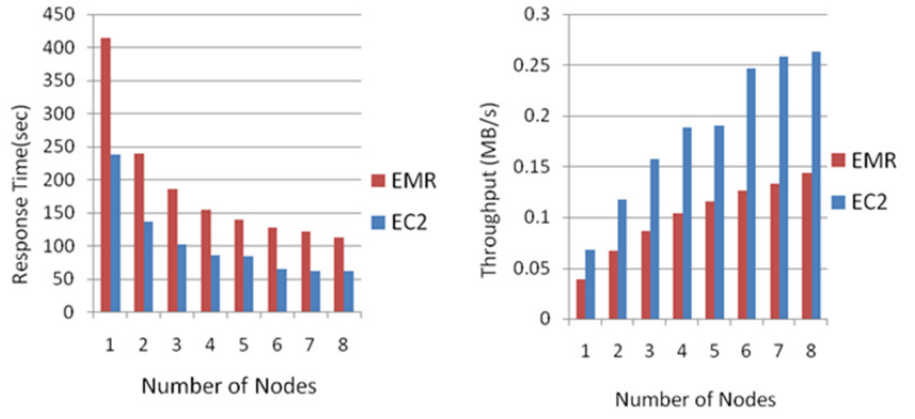

Figure 10. PageRank – Amazon EC2 vs Amazon EMR (100000 PAGES)

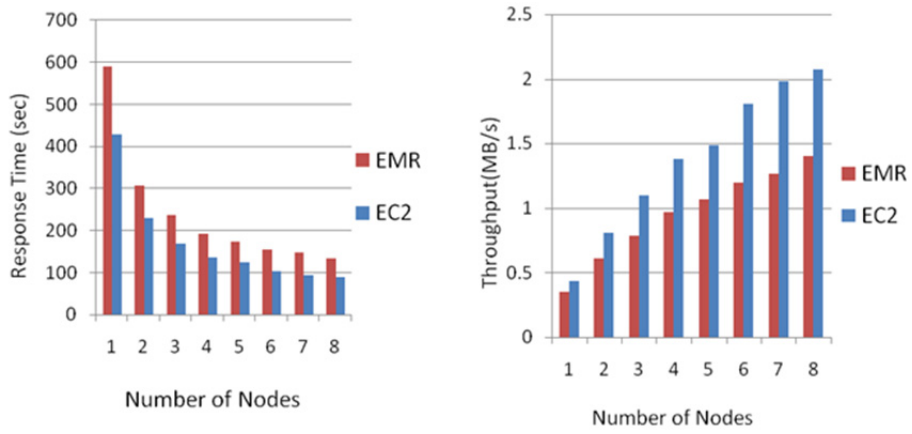

Figure 11. PageRank – Amazon EC2 vs Amazon EMR (1000000 PAGES)

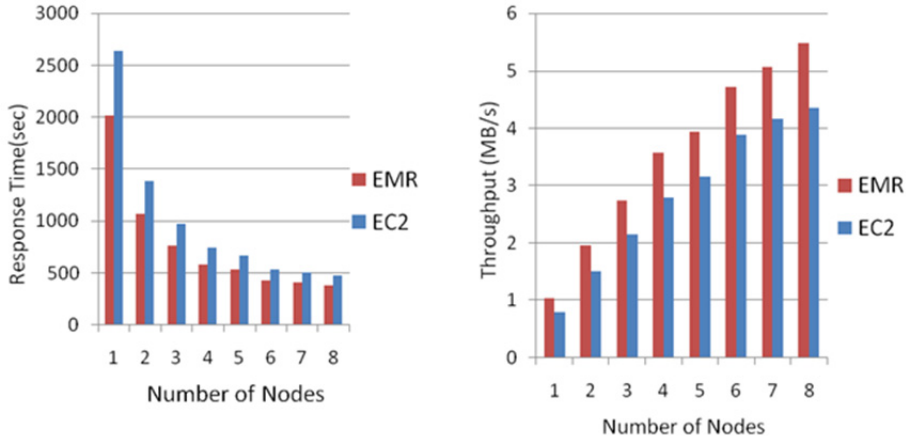

Figure 12. PageRank – Amazon EC2 vs Amazon EMR (10000000 PAGES)

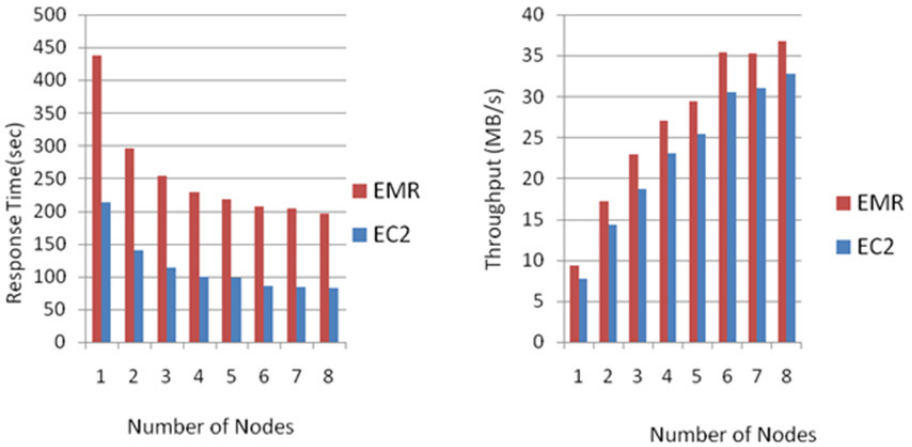

Figure 13. Hive Join – Amazon EC2 vs Amazon EMR (1000000, 600000)

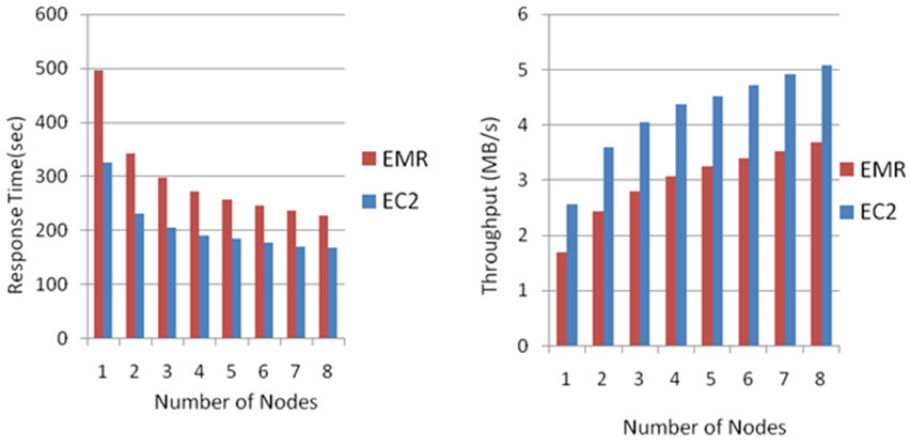

Figure 14. Hive Join – Amazon EC2 vs Amazon EMR (10000000, 6000000)

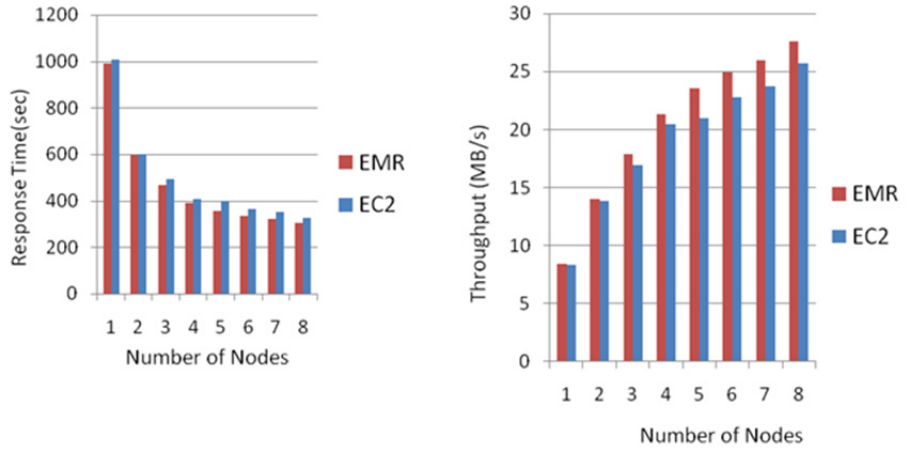

Figure 15. Hive Join – Amazon EC2 vs Amazon EMR (100000000, 60000000)

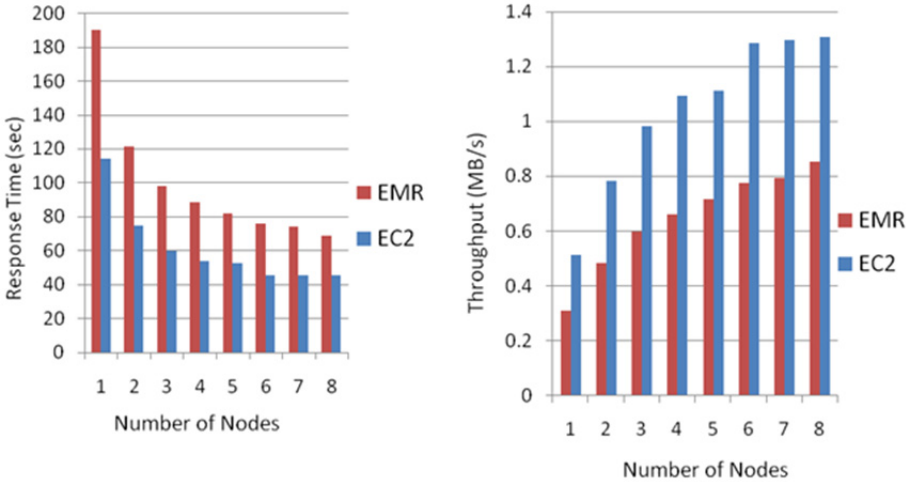

Figure 16. Hive Aggregation – Amazon EC2 vs Amazon EMR (1000000, 600000)

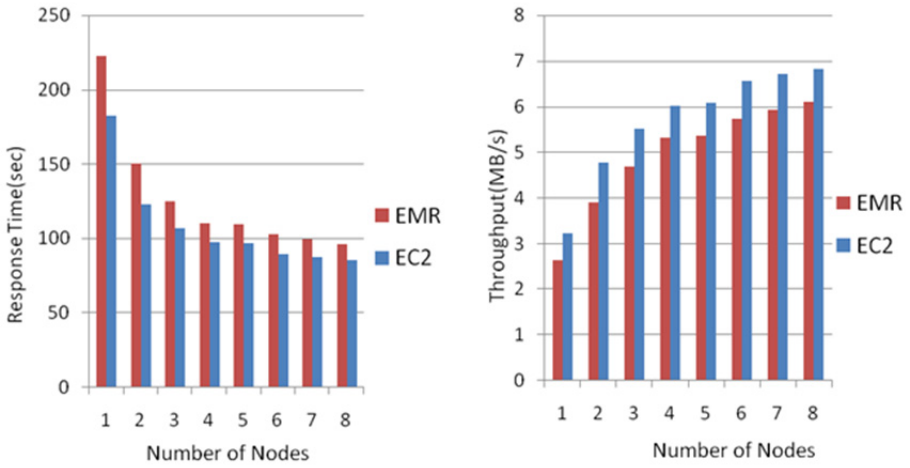

Figure 17. Hive Aggregation – Amazon EC2 vs Amazon EMR (10000000, 6000000)

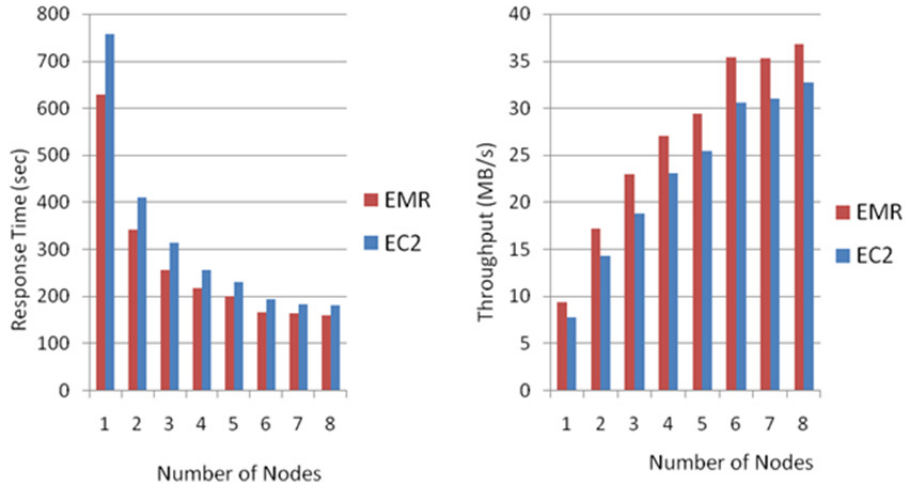

Figure 18. Hive Aggregation – Amazon EC2 vs Amazon EMR (100000000, 60000000)