DIFFERENT METHODOLOGY IN BIG DATA AND CLOUD COMPUTING RESOURCE MANAGEMENT FRAMEWORKS

[1]Mrs.J.Chinna and [2] Dr.(Mrs.) K. Kavitha

[1]Assistant Professor Department of Computer Application E.M.G Yadava women's college, Madurai, Tamil Nadu, India.

[2]Assistant Professor Department of Computer Science Mother Teresa Women's University Madurai, Tamil Nadu, India.

Abstract

Nowadays technical advancement is increased and digitizes our lives in an ultimate manner, which have led to the rapid growth of data. Such multidimensional datasets are precious due to the potential knowledge and developing decision-making insights from them. Analyzing this huge amount of data from multiple sources can help organizations to plan for the future and anticipate changing market trends and customer requirements. While the Hadoop framework is a popular platform for processing larger datasets, there are a number of other computing infrastructures, available to use in various application domains. The primary focus of the study is how to classify big data resource management systems in the context of a cloud computing environment. We identify some key features which characterize big data frameworks as well as their associated challenges and issues. We use various evaluation metrics from different aspects to identify usage scenarios of these platforms. The study came up with some interesting findings which are in contradiction with the available literature on the Internet.

1. INTRODUCTION

We live in the information age, an important measurement of present times is the amount of data that is generated everywhere around us. Data is becoming increasingly valuable. Enterprises are aiming at unlocking data's hidden potential and deliver competitive advantage [1]. Statistics MRC projected that the data analytics and Hadoop market, which accounted for $8.48 billion in 2015, is expected to reach at $99.31 billion by 2022 [2]. The global big data market has estimated that it will jump from $14.87 billion in 2013 to $46.34 billion in 2018 [3]. Gartner has predicted that data will grow by 800 percent over the next five years and 80 percent of the data will be unstructured (e-mails, documents, audio, video, and social media content) and 20 percent will be structured (e-commerce transactions and contact information) [1].

Today's largest scientific institution, CERN, produces over 200 PB of data per year in the Large Hadron Collider project (as of 2017). The amount of generated data on the Internet has already exceeded 2.5 Exabyte's per day. Within one minute, 400 hours of videos are uploaded on YouTube, 3.6 million Google searches are conducted worldwide each minute of every day, more than 656 million tweets are shared on Twitter, and more than 6.5 million pictures are shared on Instagram each day. When a dataset becomes so large that its storage and processing become challenging. Due to the constraints of existing tools and resources, the dataset is referred to as big data [4, 5]. It is the first part of the journey towards delivering decision-making insights. But instead of focusing on people, this process utilizes a much more powerful and evolving technology, given the latest breakthroughs in this field, to quickly analyze huge streams of data, from a variety of sources, and to produce one single stream of useful knowledge [6].

In this paper, we give an overview of some of the most popular and widely used big data frameworks, in the context of a cloud computing environment, which are designed to cope with the above-mentioned resource management and scaling problems. The primary object of the study is how to classify different big data resource management systems. We use various evaluation metrics for popular big data frameworks from different aspects. We also identify some key features which characterize big data frameworks as well as their associated challenges and issues. We restricted our study selection criteria to empirical studies from existing literature with reported evidence on performance evaluation of big data resource management frameworks. To the best of our knowledge, thus far there has been no empirical based performance evaluation report on major resource management frameworks. We investigated the validity of existing research by performing a confirmatory study. For this purpose, the standard performance evaluation tests as well as custom load test cases were performed on a 10+1 nodes t2.2xlarge Amazon AWS cluster. For experimentation and benchmarking, we followed the same process as outlined in our earlier study [7].

The study came up with some interesting findings which are in contradiction with the available literature on the Internet. The novelty of the study includes the categorization of cloud-based big data resource management frameworks according to their key features, comparative evaluation of the popular big data frameworks, and the best practices related to the use of big data frameworks in the cloud. The inclusion and exclusion criteria for relevant research studies are as follows: (i) we selected only those resource management frameworks for which we found empirical evidence of being offered by various cloud providers.

(ii) Several vendors offer their proprietary solutions for big data analysis which could be the potential candidate for comparative analysis being conducted in this study. However, these frameworks were not selected based on two reasons. Firstly, most of these solutions are the extension of the open-source solution and hence these exhibits the identical perform results in most of the cases. Secondly, for our empirical studies, researchers mostly prefer open-source solutions as the documentation, usage scenarios, source code, and other relevant details are freely available. Hence, we selected open-source solutions for the performance evaluation.

(iii) We did not include the frameworks which are now deprecated or discounted, such as Apache S4, in favor of other resource management systems.

. Big Data Resource Management Frameworks is offering new emerging trends and opportunities to unearth operational insight towards data management. The most challenging issues for organizations are often that the amount of data is massive which needs to be processed at an optimal speed to synthesize relevant results. Analyzing such huge amount of data from multiple sources can help organizations plan for the future and anticipate changing market trends and customer requirements. In many of the cases, big data is analyzed in batch mode. However, in many situations, we may need to react to the current state of data or analyze the data that is in motion (data that is constantly coming in and needs to be processed immediately). These applications require a continuous stream of unstructured data to be processed. Therefore, data is continuously analyzed and cached in memory before it is stored on secondary storage devices. Processing streams of data works are done by the filtering and stored in memory tables across the cluster of servers. Any delay in the data analysis can seriously impact customer satisfaction or may result in project failure [8].

While the Hadoop framework is a popular platform for processing huge datasets in parallel batch mode using commodity computational resources, there are number of other computing infrastructures that can be used in various application domains. The primary focus of this study is to investigate popular big data resource management frameworks which are commonly used in a cloud computing environment. Most of the popular big data tools available for cloud computing platform, including the Hadoop ecosystem, are available under open-source licenses. One of the key appeals of Hadoop and other open-source solutions is the low total cost of ownership. While proprietary solutions have expensive license fees and may require more costly specialized hardware, these open-source solutions have no licensing fees and can run on industry-standard hardware.

2. BIG DATA RESOURCE MANAGEMENT FRAMEWORKS

2.1 Hadoop.

Hadoop [9] is a distributed programming and storage infrastructure based on the open-source implementation of the Map-Reduce model [10]. Map Reduce is the first and current facto programming environment for developing data-centric parallel applications for parsing and processing large datasets. TheMap-Reduce is inspired by Map and Reduce primitives used in functional programming. In Map Reduce programming, users only have to write the logic of Mapper and Reducer while the process of shuffling, partitioning, and sorting is automatically handled by the execution engine. The data can either be saved in the Hadoop file system as unstructured data or in a database as structured data [11]. Hadoop Distributed File System (HDFS) is responsible for breaking large data files into smaller pieces known as blocks. The blocks are placed on different data nodes, and it is the job of the Name Node to notice what blocks on which data nodes make up the complete file. The Name Node also works as a traffic cop, handling all access to the files, including reads, writes, creates, deletes, and replication of data blocks on the data nodes. A pipeline is a link between multiple data nodes that exists to handle the transfer of data across the servers. A user application pushes a block to the first data node in the pipeline. The data node takes over and forwards the block to the next node in the pipeline; this continues until all the data, and all the data replicas, are saved to disk. Afterward, the client repeats the process by writing the next block in the file [25].

The two major components of Hadoop MapReduce are job scheduling and tracking. The early versions of Hadoop supported limited job and task tracking system. In particular, the earlier scheduler could not manage non-Map Reduce tasks and it was not capable of optimizing cluster utilization. So, a new capability was aimed at addressing these shortcomings which may offer more flexibility, scaling, efficiency, and performance. Because of these issues, Hadoop 2.0 was introduced. Alongside earlier HDFS, resource management, and MapReduce model, it introduced a new resource management layer called Yet Another Resource Negotiator (YARN) that takes care of better resource utilization [12].

3. DIFFERENT METHODOLOGY USING MAP REDUCING TECHNIQUES

Veiga et al. [13] conducted a series of experiments on a multicore cluster setup to demonstrate performance results of Apache Hadoop, Spark, and Flink. Apache Spark and Flink resulted to be much efficient execution platforms over Hadoop while performing nonsort benchmarks. It was further noted that Spark showed better performance results for operations such asWordCount and K-Means (CPU-bound in nature) while Flink achieved better results in the PageRank algorithm (memory bound in nature).

Mavridis and Karatza [14] experimentally compared performance statistics of Apache Hadoop and Spark on Okeanos IaaS cloud platform. For each set of experiments, necessary statistics related to execution time, working nodes, and the dataset size were recorded. Spark performance was found optimal as compared to Hadoop for most of the cases. Furthermore, Spark on YARN platform showed suboptimal results as compared to the case when it was executed in standalone mode. Some similar results were also observed by Zaharia et al. [15] on a 100 GB dataset record.

Vellaipandiyan and Raja [16] demonstrated performance evaluation and comparison of Hadoop and Spark frameworks on resident’s record dataset ranging from 100 GB to 900 GB of size. Spark scale of performance was relatively better when the dataset size was between small and medium size (100 GB–750 GB); afterward, its performance declined as compared to Hadoop. The primary reason for the performance decline was evident as Spark cache size could not fit into the memory for the larger dataset.

Taran et al. [17] quantified performance differences of Hadoop and Spark using WordCount dataset which was ranging from 100 KB to 1 GB. It was observed that the Hadoop framework was five times faster than Spark when the evaluation was performed using a larger set of data sources. However, for the smaller tasks, Spark showed better performance results. However, the speed-up ratio was decreased for both databases with the growth of input dataset.

In the area of big data, there is still a clear gap that requires more effort from the research community to build an in-depth understanding of the performance characteristics of big data resource management frameworks. We consider this study a step towards enlarging our knowledge to understand the big data world and provide an effort towards the direction of improving the state of the art and achieving the big vision on the big data domain. In earlier studies, a clear ranking cannot be established as the study parameters were mostly limited to a few issues such as throughput, latency, and machine learning. Furthermore, further investigation is required on resource engines such as Apache Samza in comparison with other frameworks. In addition, research effort needs to be carried out in several areas such as data organization, platform -specific tools, and technological issues in the big data domain in order to create next-generation big data infrastructures.

This paper analysis distributed graph processing environment and represents an efficient resource management mechanisms on the aspect of cost as well as system performance by means of different partitioning and scheduling techniques.

Challenges in large-scale graph processing on cloud environments

In cloud computing, distributed computing is the process of applying unprecedented computing capabilities which are available in the cloud environments. The main advantages of this distributed computing are flexibility and pay-as-you-go pricing models. However, these capabilities have increased the interests among the research community towards graph processing applications, the real-time issues, and research questions still exist. We discuss those issues and represented the computing paradigm in order to enhance the approach of large-scale graphs.

In the below section, we discuss the most commonly used algorithm and their demonstrations along with its studies on large graph processing.

Graph Traversal Algorithms: In this methodology, in order to examine or update the vertices values by means of the procedure the algorithm travels on all vertices of the graph [199]. Mostly the common algorithms are of Breadth-First-Search (BFS) and Depth-First-Search (DFS) type [18]. These two algorithms follow traverse of graph tree for finding a particular node. Otherwise, it will visit all nodes in a certain order. In the graph, for finding the minimum path among the particular node and any arbitrary node the Single Source Shortest Path (SSSP) is implemented on the aspect of minimum cost or weight [19]. In this type, the most famous algorithms are Dijkstra’s algorithm and the Bellman-Ford algorithm [20].

Graph Analysis Algorithms: These type of algorithms checks the graph's topology for identifying the graph objects and its complexities. Most of the protein interplay analysis and social network analysis [21] has these graph statistics and topological measures.

Google’ Pregel is a kind of distributed vertex-centric framework that implies master-worker architecture on several hosts of a cluster. At the Carnegie Mellon University, the GraphLab is developed and later reinforced by GraphLab Inc., for developing single machine processing . But the outcome as a distributed one [22]. The other Pregel-like systems which are developed as a distributed graph processing systems are GPS [23], Mizan and GoFFish
[208]. In this way, some of the non-Pregel-like systems are Presto [24], Trinity [25], and Surfer [46]. Frameworks like GraphX are structured on top of Spark distributed

dataflow system. For their execution environment, all systems utilize the multi-node clusters or cloud VMs. As in which none of them achieves elasticity property of Clouds, instead of that its treat captive VMs as a commodity cluster.

Consider too the above-mentioned graph frameworks, there are various graph processing libraries are developed to achieve high-performance computing (HPC) clusters. Boost graph library (BGL) is a kind of generic graph processing library that offers generic interfaces between the graph's structure and common operations. But in this mechanism, the implementation details are hidden. This feature makes the BGL implementation on graph algorithms on the shared-memory and parallel computing platforms. Graph500 is a type of graph processing benchmark based on several metrics. On this the supercomputers, different matrices are measured such as memory size for graph storage, communication performance, and performance of random access to memory. It differs from Top500 , as it is structured for compute-intensive applications. To achieve the high-performance computing by means of parallel graph frameworks several attempts are done such as MPI , PVM , BLAS , JUNG and LEAD . But on which no one is a success in achieving the general-purpose graph processing platforms flexibility .

Pattern Matching: These algorithms were utilized in the graph to find the occurrence of input patterns, on the finding it may be an exact or approximate recognition .

Graph Anonymization: These algorithms were applicable to creating a new graph by means of its original graph. On which latter reveals its specific topological or attribute properties. By means of it, intruders to re-identify the network can be prevented .

REFERENCE

[1] "Big Data in the Cloud: Converging Technologies, (August 11)," https://www.intel.com/content/dam/www/public/emea/de/de/ documents/product-briefs/big-data-cloud-technologies-brief .pdf.

[2] Q. He, J. Yan, R. Kowalczyk, H. Jin, and Y. Yang, "Lifetime service level agreement management with autonomous agents for services provision," Information Sciences, vol. 179, no. 15, pp. 2591–2605, 2009.

[3] O. Rana, M. Warnier, T. B. Quillinan, and F. Brazier, "Monitoring and reputation mechanisms for service level agreements," in 5th International Workshop on Grid Economics and Business Models, GECON '08, vol. 5206 of Lecture Notes in Computer Science (including subseries Lecture Notes in Artifcial Intelligence and Lecture Notes in Bioinformatics): Preface, pp. 125–139, 2008.

[4] N. Yuhanna and M. Gualtieri, "Te Forrester Wave: Big Data Hadoop Cloud Solutions, Q2 2016 Elasticity, Automation, and Pay-As-You-Go Compel Enterprise Adoption of Hadoop in the Cloud, (June 20)," https://ncmedia.azureedge.net/ncmedia/ 2016/05/Te Forrester Wave Big D.pdf.

[5] R. Arora, "An introduction to big data, high performance computing, high-throughput computing, and Hadoop," Conquering Big Data with High Performance Computing, pp. 1–12, 2016.

[6] F. Pop, J. Kołodziej, and B. D. Martino, Resource Management for Big Data Platforms Algorithms, Modelling, and High-Performance Computing Techniques, Springer, 2016.

[7] S. Ullah, M. D. Awan, and M. S. Khiyal, “A price-performance analysis of EC2, google compute and rackspace cloud providers for scientifc computing,” Journal of Mathematics and Computer Science, vol. 16, no. 02, pp. 178–192, 2016.

[8] J. Hurwitz, A. Nugent, F. Halper, and M. Kaufman, Big Data for Dummies, John Wiley Sons, 2013.

[9] S. K. Garg, S. Versteeg, and R. Buyya, “A framework for ranking of cloud computing services,” Future Generation Computer Systems, vol. 29, no. 4, pp. 1012–1023, 2013.

[10] D. Wu, S. Sakr, and L. Zhu, “Big data programming models,” in Handbook of Big Data Technologies, pp. 31–63, 2017

[11] “Big data working group big data taxonomy, 2014”.

[12] J. Hurwitz, A. Nugent, F. Halper, and M. Kaufman, Big Data for Dummies, John Wiley Sons, 2013.

[13] J. Veiga, R. R. Exposito, X. C. Pardo, G. L. Taboada, and J. Tourifo, “Performance evaluation of big data frameworks for large-scale data analytics,” in Proceedings of the 4th IEEE International Conference on Big Data, Big Data 2016, pp. 424–431, December 2016.

[14] I. Mavridis and H. Karatza, “Log fle analysis in cloud with Apache Hadoop and Apache Spark,” in Proceedings of the Second International Workshop on Sustainable Ultrascale Computing Systems (NESUS 2015), Poland, 2015.

[15] M. Zaharia, M. Chowdhury, and T. Das, “Fast and interactive analytics over Hadoop data with Spark,” USENIX Login, vol. 37, no. 4, pp. 45–51, 2012.

[16] S. Vellaipandiyan and P. V. Raja, “Performance evaluation of distributed framework over YARN cluster manager,” in Proceedings of the 2016 IEEE International Conference on Computational Intelligence and Computing Research, ICCIC 2016, December 2016.

[17] V. Taran, O. Alienin, S. Stirenko, Y. Gordienko, and A. Rojbi, “Performance evaluation of distributed computing environments with Hadoop and Spark frameworks,” in Proceedings of the 2017 IEEE International Young Scientists’ Forum on Applied Physics and Engineering (YSF), pp. 80–83, Lviv, Ukraine, October 20

[18] D. C. Kozen, "Depth-First and Breadth-First Search," in The Design and Analysis of Algorithms. New York, NY. US: Springer New York, 1992, pp. 19-24.

[19] P. Roy, "A new memetic algorithm with GA crossover technique to solve Single Source Shortest Path (SSSP) problem," in Proceedings of the 2014 Annual IEEE India Conference (INDICON), Pune, India, 2014

[20] M. J. Bannister and D. Eppstein, "Randomized Speedup of the Bellman-Ford Algorithm," In Proeedings of the Meeting on Analytic Algorithmics and Combinatorics (ANALCO'12, Kyoto, Japan, 2012, pp. 41-47.

[21] H. K. Lau, "Error Detection in Swarm Robotics: A Focus on Adaptivity to Dynamic Environments," York, UK, PhD Thesis2012.

[22] Y. Low et al., "Distributed GraphLab: A Framework for Machine Learning and Data Mining in the Cloud," VLDB Endowment, vol. 5, no. 8, pp. 716-727, 2012

[23] S. Salihoglu and J. Widom, "GPS: A Graph Processing System," in Proceedings of the 25th International Conference o Scientific and Statistical Database Management (SSDBM), Baltimore, Maryland, 2013.

[24] S. Venkataraman, E. Bodzsar, I. Roy, A. AuYoung , and R. S. Schreiber, "Presto: Distributed Machine Learning and Graph Processing with Sparse Matrices," in Proceedings of the 8th ACM European Conference on Computer Systems (EuroSys '13), Prague, Czech Republic, 2013, pp. 197-210.

[25] B. Shao, H. Wang, and Y. Li, "Trinity: A Distributed Graph Engine on a Memory Cloud," in Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '13), New York, USA, 2013, pp. 505-516.