An In-Memory-Based Big Data Analytics with Two-Level Storage on Private Cloud

Nikkita Shekhar and Ambika Pawar

Abstract With growing capacity of main memory, in-memory big data management and processing is developing and being used in many big data applications. It supports interactive data analysis by improving I/O throughput. Memory-centric distributed file systems such as tachyon and in-memory data clustering framework like Apache Spark are being used in analytical problems where both speed and fault tolerance are mandatory. In order to achieve high-speed big data processing, we proposed a system design which involves two-tier storage architecture which is the combination of HDFS and in-memory-based file system tachyon. Also, our architecture involves Apache Spark, an open-source in-memory-based data processing tool to analyse the big data. In this framework we would utilise the main memory by integrating caching algorithm to improve the data processing time. As the experimental result, we would demonstrate the comparison between performance of traditional Hadoop MapReduce and this in-memory-based framework. In this paper, we survey the existing storage and computation infrastructures, their performance while integrating together and contribution of such infrastructures in solving many I/O intensive analytical issues.

Keywords Big data · Analytics · In-memory · Spark · Hadoop · Parallel computing · Distributed system · I/O throughput · Computation speed

1 Introduction

In this digital era, society is becoming more instrumented and as a result of that, a large amount of data is getting generated each and every second. Today, the main sources of data are electronic devices, social network, business data, sensors, stock

Nikkita Shekhar (✉) · Ambika Pawar
Department of Computer Science, SIT, Symbiosis International University, Pune, India
e-mail: nikkita.shekhar@sitpune.edu.in

Ambika Pawar
e-mail: ambikap@sitpune.edu.in

R. Singh and S. Choudhury (eds.), *Proceeding of International Conference on Intelligent Communication, Control and Devices*, Advances in Intelligent Systems and Computing 479, DOI 10.1007/978-981-10-1708-7_109

markets, etc. This large volume of data does not make any sense until it gets analysed and give some informative output. With this explosion of big data, the requirements of efficient solution to deal with the aspects like data storage, real-time processing and information reorganisation have become mandatory. Big data analytics is the field of research to uncover hidden pattern and information from a large volume of random data in structure, unstructured or semi-structure format. This is usually accomplished through implementation of large data storage and high-performance computation engine with suitable iterative algorithm. To perform any analysis on such huge amount of complex data, a suitable hardware and software platform is required. Two things are need to be considered while choosing the right resources: (i) How big is the data to be processed? and (ii) How quickly the data need to be processed?

In private Cloud where storage and computation are done at separate machine, to perform analysis on big data, following points are need to be kept in mind: system scalability, CPU and computing, data volume and storage and network bandwidth. This system should be elastic to scale up and scale out as per the requirement. It should be capable to handle huge data in storage and computation engine should be strong enough to compute the data processing in time. Although there are many successful traditional tools and technology available to perform the same [1], with the growth in volume and velocity of data generation, a faster and reliable system is required. With growing capacity of main memory, in-memory big data management and processing is developing and being used in many big data applications. It supports interactive data analysis by improving I/O throughput [2].

In analytical system, RAM can be used as data storage as well as temporary location for intermediate result of data computation process. Using RAM as middleware removes I/O overhead between the machines. This improves the speed of analysis as the time taken by the I/O requests has been removed. Also, for an iterative algorithm, keeping the intermediate result in main memory enhances the overall performance of analytical engine.

In memory, feature can be implemented in an analytical project as IMDB (in-memory database) and in-memory-based analytical engine. Memory-centric distributed file systems such as tachyon and in-memory data clustering framework like Apache Spark are being used in analytical problems where both speed and fault tolerance are mandatory. In this paper, we are proposing a system design for big data analytics problem utilising in-memory features as storage as well as in analytical engine in order to improve the data computation speed. In Sect. 2, we have given literature review of the related components. In Sect. 3, proposed idea and system design is discussed, and Sect. 4 gives a brief detail about experimental setup and preliminary results.

2 Literature Review

In this section, we will give brief introduction about our system component and literature survey about them. First, we will discuss about Hadoop and its limitation in big data analytics, importance of in-memory technology and existing works on them.

2.1 Apache Hadoop

Apache Hadoop is an open-source framework for storing and processing big data sets using clusters. Hadoop platform contains two major components: 1. Distributed file system, and 2. Hadoop YARN that manages the job schedules across cluster. MapReduce is a programming model in Hadoop that breaks the task into small parts and process them in parallel. One major drawback of Hadoop MapReduce is its inefficiency in running iterative processes [3]. After each iteration, data is read/write to disc which increases i/o overhead and degrades the performance [4].

2.2 Apache Spark

Apache spark is a next-generation in-memory-based paradigm developed by researchers at university of California at Berkeley. It is developed to overcome the i/o limitation and an alternative for Hadoop MapReduce. Spark allows the data to be cached in memory and thus removes disc overhead and increases the processing speed by 100× [5]. Also, it generates RDD and DAG graph to keep track of job scheduling [3].

2.3 Tachyon

For distributed storage, the read throughput can be improved using caching; however, the write throughput is limited by both disc and network bandwidth due to data replication for fault tolerance. Tachyon is a memory-centric, fault tolerant distributed file system which enables reliable data exchange at in-memory speed across cluster frameworks [6]. Tachyon uses a master–slave architecture similar to other cluster file systems, where each worker manages local blocks and shares them with applications through a RAMFS. Files in Tachyon are organised in a tree hierarchy and identified by their paths. Hence, it provides fault tolerance in case of system failure [7].

2.4 Survey

Various research approaches are proposed by scholars and researchers for in-memory utilisation in big data analytics.

In Xuan et al. [7], on a HPC cluster, two-level storages with parallel file system OrangeFS and in-memory file system Tachyon are integrated together to enhance the I/O throughput and fault tolerance for a scalable system. They characterise the I/O behaviour of proposed system and compare the performance with HDFS and OrangeFS using TeraSort benchmark.

Jorge et al. [8] utilised the memory to speed up the computation by integrating Spark on Hadoop. They implemented iterative algorithm like KNN and Pegasus SVM on Spark framework and compare its performance with MPI. As a result, they conclude that Spark give speed but MPI is more reliable for fault tolerance.

Yan et al. [9] researcher suggested a domain-specific seismic analytical Cloud as a service. They used Spark and YARN to create a cloud platform for the processing of seismic dataset which is very complex to deal with. SAC, seismic analytical cloud, as a service provides simple and better platform to work on seismic data.

Khan et al. [3] demonstrated the effectiveness of the analytics service for future smart city implementation. They identify correlation between different environmental indicators of city datasets using Hadoop and Spark. They compare the result on the basis of computation speed between Hadoop MapReduce and Apache Spark.

Recently, during our research, we found Baidu, a search engine in china, and also used Tachyon and collaborated with research team of TachyonFS [10]. They have achieved 30× to 60× speedups. They are also working on some limitations like block size issue, memory swapping, etc.

3 Proposed Idea

In order to achieve high-speed big data processing, this framework would involve two-tier storage architecture which is the combination of HDFS and in-memory-based file system Tachyon as shown in Fig. 1. Also, our design involves Apache Spark, an open-source in-memory-based data processing tool to analyse the big data. In this framework we would utilise the main memory by integrating caching algorithm to improve the read/write request miss and hit ratio [7]. As the experimental result, we would demonstrate the comparison between performance of traditional Hadoop MapReduce and proposed in-memory-based framework. Our system design involves integrating main memory features integration with tradition

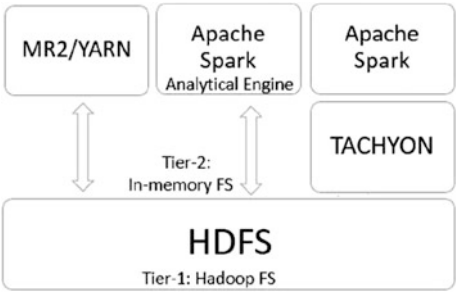


Fig. 1 System design of two-level storage framework

data processing techniques. Such framework would contribute in solving many I/O intensive analytical issues.

4 Proposed Architecture

In the proposed architecture, we describe our framework for two-level storage system in which we have integrated an in-memory file system like Tachyon at name node on the top of traditional distributed file system HDFS at data node of the distributed cluster. Also, we have combined in-memory-based computation engine which is supposed to enhance the aggregate I/O throughput efficiently for an iterative algorithm (Fig. 2).

We implemented a prototype of two-level storage system by integrating tachyon-7.1.0 with HDFS-2.6.1 on Apache Spark-1.5.0. Here HDFS works as disc

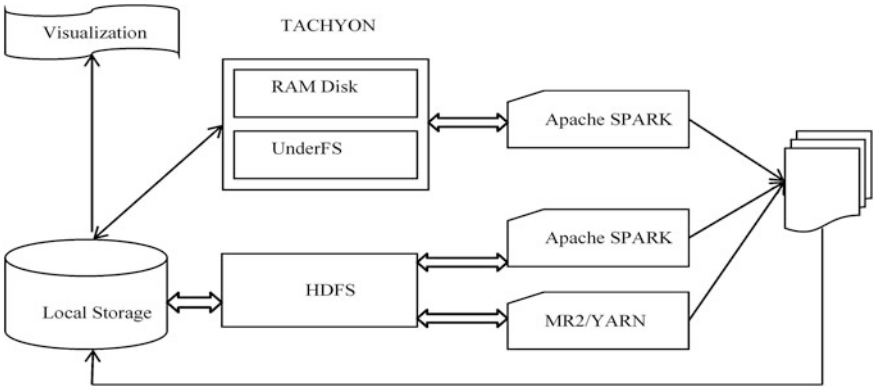


Fig. 2 System architecture of proposed framework

storage where data is stored in distributed manner, and Tachyon acts like cache memory between disc and analytical engine Spark. In this architecture, a set of fix-sized logical blocks are considered for input file in Tachyon. It controls parallel granularity and also RDDs generated by analytical engine Spark are managed here. In order to enhance the I/O throughput and speed up the computation, Spark and Tachyon work together at compute node while input files are distributed over Hadoop file system in cluster. Tachyon manages the cached file from HDFS as well as the intermediate output result from spark. In our system, we have removed the read/write overhead of reading data from remote data nodes, especially when the nodes are overloaded with request, by applying priority policy. A cache I/O buffer is applied between Spark and Tachyon to set the priority of data that are being stored in Tachyon from different sources, i.e., Spark RDD [4] and input file from disc.

5 Experimental Evaluation and Preliminary Result

In this section, we are evaluating prototype for our two-level storage-based analytical setup and further we are comparing it with the traditional Hadoop MapReduce. For this, two experimental setups are established: First, traditional Hadoop MapReduce, and another is a two-layer storage architecture with Tachyon over HDFS. In private cloud where resources are limited [11], we have used the following configuration as shown in Table 1. We have setup a five-node Hadoop cluster, each with 2 GB RAM and 1 TB storage. As Apache Spark and Tachyon are memory sensitive, 8 GB of memory is allocated for both at master node.

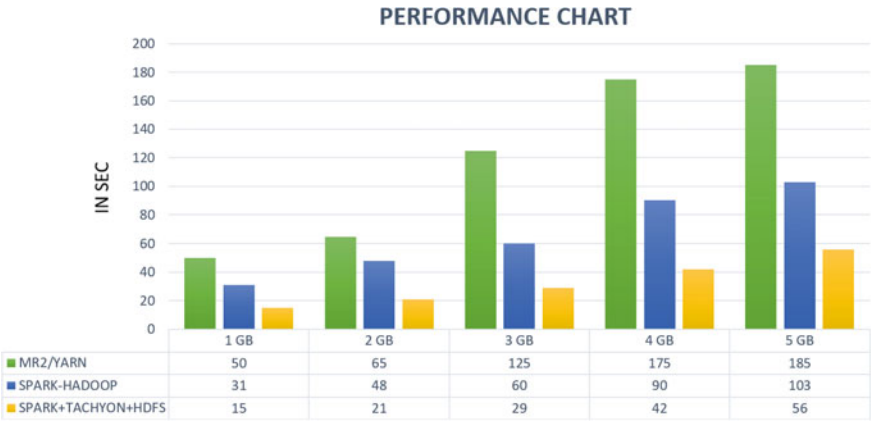
We have chosen anomaly detection problem for Wikipedia datasets of different sizes. We used supervised learning technique to find the outliers. We used 10 MB of cache buffer between Spark and Tachyon on compute node.

Three experiments are performed using traditional Hadoop MapReduce, Spark on Hadoop and out two-level storage system with HDFS, Tachyon and Apache Spark. We compared our tentative result in the form of graph as shown in Graph 1. Here we can see that our proposed system gives high-speed performance for particular problem.

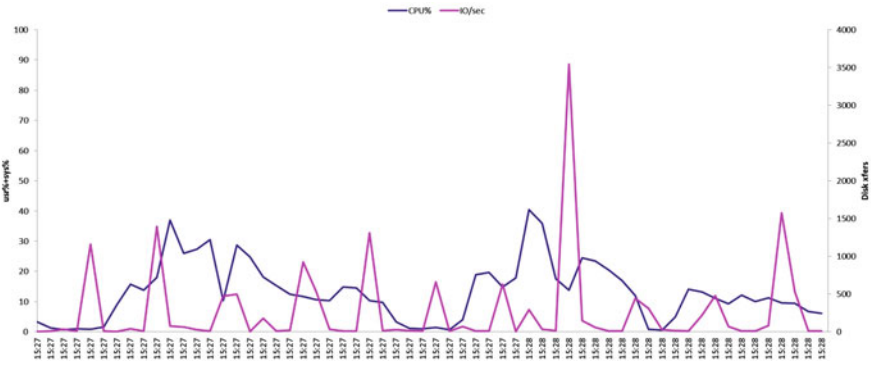
In other Graph 2 we can see that the comparison between read and write operation time with integrated file systems is used in our framework.

Table 1 Experimental setup configuration

Master node			
CPU	Memory	Storage	Network
Intel i7 2.5 GHz	8 GB	1 TB	10 MBPS
Data node			
CPU	Memory	Storage	Network
Intel i5 2.5 GHz	2 GB	1 TB	10 MBPS



Graph 1 Performance analysis



Graph 2 CPU I/O analysis for proposed design

6 Conclusion

With a brief discussion we have proposed an experimental framework for big data analytics especially for iterative problems. To make our framework efficient we have used the caching concept in two ways. First, we use i/o cache buffer between analysis process and main memory for data read operation and also providing caching for data write operation using TachyonFS. We have used in-memory features to remove I/O request overhead, increase fault tolerance and finally improve the speed of data processing. We compared our result with traditional Hadoop. In future, we are looking forward to make our framework more suitable for real-time problems in a small private Cloud environment where resources may be limited.

References

1. Dilpreet Singh and Chandan K Reddy: A survey on platforms for big data analytics. Singh and Reddy Journal of Big Data 2014, 1:8.
2. Hao Zhang, Gang Chen, Member, IEEE, Beng Chin Ooi, Fellow, IEEE, Kian-Lee Tan, Member, IEEE, and Meihui Zhang, Member, IEEE: In-Memory Big Data Management and Processing: A Survey. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 27, NO. 7, JULY (2015).
3. Zaheer Khan, Ashiq Anjum, Kamran Soomro and Muhammad Atif Tahir: Towards cloud based big data analytics for smart future cities. Khan et al.; licensee Springer Journal of Cloud Computing: Advances, Systems and Applications 4:2 (2015).
4. M. J. Franklin, S. Shenker, and I. Stoica. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In NSDI, 2012.
5. Juwei Shiz, Yunjie Qiuy, Umar Farooq Minhasx, Limei Jiaoy, Chen Wang; Clash of the Titans: MapReduce vs. Spark for Large Scale Data Analytics. Proceedings of the VLDB Endowment, Vol. 8, No. 13 (2015).
6. H. Li, A. Ghodsi, M. Zaharia, S. Shenker, and I. Stoica. Tachyon: Reliable, memory speed storage for cluster computing frameworks. In SOCC, pages 1–15 2014.
7. Pengfei Xuan, Feng Luo, Pradip K Srimani: Big Data Analytics on Traditional HPC Infrastructure Using Two-Level Storage. School of Computing, Clemson University (2015).
8. Jorge L. Reyes-Ortiz, Luca Oneto, and Davide Anguita: Big Data Analytics in the Cloud Spark on Hadoop vs MPI/OpenMP on Beowulf. The Scientific Programme Committee of INNS-BigData conference (2015).
9. Yuzhong Yan, Mahsa Hanifi, Liqi Yi, Lei Huang: Building a Productive Domain-Specific Cloud for Big Data Processing and Analytics Service. Journal of Computer and Communications, 3, 107–117 (2015).
10. https://www.oreilly.com/ideas/accelerating-big-data-analytics-workloads-with-tachyon (Online).
11. Ahsan Javed Awan, Mats Brorsson, Vladimir Vlassov and Eduard Ayguade; Performance Characterization of In-Memory Data Analytics on a Modern Cloud Server. SIGMOD'15 Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data Pages 631–646 ACM New York, NY, USA (2015) ISBN: 978-1-4503-2758-9.