ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

Beyond Batch Process: A BigData processing Platform based on Memory Computing and Streaming Data

M.Jayashree, S.Zahoor Ul Huq

PG Student, Department of CSE, G.Pulla Reddy Engineering College (Autonomous), Kurnool, India

Professor Dept of CSE, G.Pulla Reddy Engineering College, Kurnool, Andhra Pradesh, India

ABSTRACT: In this research we perform analysis on large data sets of students which will be obliging smart environment. Such a persisting and spring of students data is analysed using batch analysis technique. Beyond batch process streaming data analysis is performed on the basis of word-count program that runs data from HDFS and dynamically created data. To compute such coherent strategies we use an amateur schema called batch and streaming process. This architecture reduced to serve as X-Platform where many tools can be used for batch and stream analysis on this framework. We compute the immutable data using spark-sql which is a query language where it provides the bridge for interactive process to have iterative operations too. The processing of real-time streaming data includes works using spark-streaming. We evaluate preliminary results and analysis report, where we compare performance on datasets and achieve a low-latency rate due to RDD used.

KEYWORDS: Spark, RDD, Spark SQL, interactive, Spark Streaming, in-memory and Hive.

I. INTRODUCTION

In the today's smart world, the data is being increased like a storm. Due to that rapid growth on data a wide range of technologies are emerged on to the BigData world over the past few years. Among that cluster environment bought several challenges. To overcome such challenges many frameworks have been introduced. One of them are MapReduce , introduced by Google which runs colossal applications on commodity machines as a clusters. Such frame works are introduced to collaborate with huge data sets in a scalable, reliable and fault-tolerant way.

A. BACKGROUND

The Apache Hadoop is a framework that allows for the distributed processing of large data sets across clusters of commodity computers using simple programming models[12]. Hadoop Cluster can scale up from single servers to thousands of machines and each machine has the capability of local computation and storage. Which deals with different kinds of data. This ecologic system include MapReduce, HDFS, Hive, Pig, impala, HBase and so on.

Hive is a data warehouse built on top of hadoop which provide a Query Language where queries are ad-hoc[13]. Hive splits into parallel tasks using MapReduce which can perform huge on peta bytes of data stored. It maintains a meta-store for every table created where meta data such as schema and location are stored.

The demand for new efficient methods to process both streaming data and immutable data is growing rapidly. It is one of the key challenge in Big Data world. Performing low-latency analysis with Real-Time data and historic data and is a part of it. *Spark* is a distribute and parallel computing platform for storing and analysing such data[1][2][10].

Apache Spark is a platform which supports applications with working sets while providing similar scalability and fault tolerance properties of MapReduce. Spark is implemented in Scala, the main aim of it is the *Resilient distributed dataset*(RDD)[3][8][11], which represents as in-memory cluster of objects partitioned across a set of machines that can rebuilt if a partition is lost. Due to this in-memory cluster computing users can easily cache an RDD across the machines and reuse it multiple times for parallel operations and perform interactive operations.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

B. PROBLEM STATEMENT

Hive deals with ad-hoc queries but it does not support sub queries .Mean while it does not meet the demand of Real-Time interactive query in the Big Data, where the output of a process can be a input to other process. The limitation of Map Reduce is that it persists the full dataset to HDFS after running each job. In this scenario large data sets may encounter high latency with lot of disk access, I/O's number of computations.

This can be overcome by Spark which takes a more holistic view of a pipeline of operations. When the output of an operation needs to be fed to another operation, Spark allows to access the data directly without writing to persistent storage.

The main innovation of Spark was to introduce an in-memory caching abstraction(RDD). This makes Spark ideal for workloads where multiple operations access the same input data[4][5]. Users can instruct Spark to cache input data sets in memory, so they don't need to be read from disk for each operation. Spark somehow runs entirely in-memory while MapReduce does not. Spark shuffle implementation works very similarly to MapReduce where each record is serialized and written out to disk on the map side and then fetched and de serialized on the reduce side.

C: PAPER ORGANIZATION

The rest of this paper is organized as follows. In section 2 we provide the system overview which shows the importance of spark in-memory cluster computing process. In section3, we built the proposed framework which describe the flow diagram of batch and streaming process. In section4, experimental approach which gives description about the work environment and sample datasets used for the application we built. In section5, we elaborate the batch, interactive, streaming process and queries applied. In section6, a conduct analysis is performed which shows the performance analysis of spark. In section7, we provide the survey of the related work about Hive, Spark SQL and Spark streaming. At last, we explain the impact of different query tools on real-time data and spark SQL is the best choice for it which shows the low latency rate and high performance.

II. RELATED WORK

Continuous high speed data streams generate large data volume it has become an challenge in BigData.

In[1], streaming analysis is performed on traffic data where it has used spark as a platform for low latency rate and data injection tool has Flume for streaming data. In[2], Zhijie Han used spark has an primary framework and point out the core technologies porn's and corn's. Li Gu [3], introduces memory usage computations and running time comparison over hadoop on iterative operations using page rank algorithm. Xiaopeng Li [4], different comparison are made on parquet file format for the quick query speed on Hive, Impala and SparkSQL. Consequently Impala is more suitable for parquet. In[5], shows the overview of spark over MapReduce frameworks and compare the various parameters followed by some analysis using K-means algorithm. In[6] real-time streaming data is considered using twitter where spark streaming is the best to choose and do applications on streaming data. Effective scheduling strategy[7] are applied to reduce the worst case event processing time by scheduling algorithm on spark streaming. In[8] it provides the overall introduction to SparkSQL and the importance of RDD. Lambda Architecture [9] describes both batch and speed data processing in a cost-effective way using spark on cloud. This paper[10] presents the performance prediction of jobs running on Apache Spark such as execution time, memory and I/O cost. RDD[11] an effective and fault-tolerant abstraction for sharing data in cluster application. Based on the characteristics of BigData[12] different algorithms and solutions are proposed for applications. Scientific Data Management[13] application need a low cost, scalable, easy-to-use and fault-tolerance platform, also need a data warehouse to built it. so Hive is best suitable for it and evaluated the performance analysis on it.

III. SYSTEM OVERVIEW

As a general framework we just provide Hadoop for building different types of BigData applications for analysis where it provides an open source java implementation of MapReduce. It is composed of two phases: Hadoop Distributed File System(HDFS) for data storage and MapReduce for data processing. To process job's it has scheduled into number of tasks and many algorithms are placed to run it. And also for iterative operations different MapReduce jobs can't keep

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

and share data frequently where it has to store data in HDFS and write back to it again, which required a lot of disk access, I/Os and number of computations.

Apache Spark is a cluster computing frame work which is designed to overcome Hadoop storage issue for iterative operations[7]. Spark is compactable with Hadoop and uses all the properties of it(e.g. HDFS, HBase). Spark introduces a new concept called RDD(*Resilient Distributed Dataset)* to cache data. The main motive of RDD is to build application that currently computing frame works handle inefficiently: iterative algorithms and interactive data mining tools. In both the cases keeping data in memory can rapidly increase performance. RDD is read-only, partitioned collection of records. It is created for *persistence* storage where users can reuse and choose a storage strategy for them. Spark keeps this RDDs persistent in memory by default , but it can also split them to disk if it is not fit in RAM[1][3]. If any changes in current RDDs will create new RDDs. We have implemented RDDs in a system that outperforms Hadoop by up to 20x in iterative applications and can be used for different interactive queries on gigabytes of data.

IV. PROPOSED FRAMEWORK

The proposed system consist of batch, interactive and streaming process. In scenarios such as speed and time computation to minimize latency rate, where output is fed as input of other data . we describe it our proposed architecture. It consists of two parts batch process for the persistence data and interactive along streaming process for iterative operations. All these are built on a single platform called Apache Spark with in-built API's for programming in Java , Python and Scala. Spark also provide a DAG visualization of the operations performed and flow to analyse better.

A. BATCH PROCESS

In figure 1, narrate about batch process, firstly incoming data may be of different type where the data is stored persistently. The immutable data stored act as an input for any X-platform like Hadoop or spark, where the data is divided into number of chunks to perform computations as required and served as an output. In this paper we are presenting SparkQL and HiveQL to process immutable data. Here we perform different query operation on the data set we considered to make an analysis report. If any failure in the computation process it re computes the process again and serve the output as required.

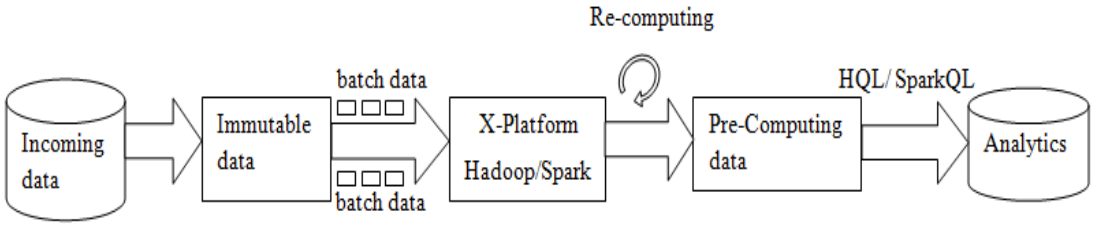


Figure 1. Batch Process

B. INTERACTIVE & STREAMING PROCESS

Beyond Batch process we are moving towards interactive and streaming data analysis. Interactive operations are performed on immutable data using Spark SQL, the major role in this process is in-memory data where the data can be stored as an RDD to perform different transformations and actions. On static data we perform different query operations where the data is stored on RDD, in the form of objects. On those resultant data we can have different interactive session by performing several actions and transformations. The RDD stored is persistent where if users want to perform any actions on it, the current RDD won't change but it creates different RDD for it.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

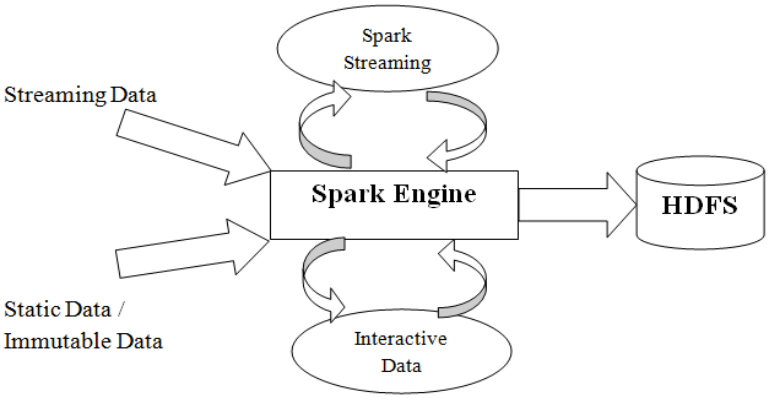


Figure 2. interactive & Streaming process

Real-time data deals with the streaming process where data is injected from the HDFS, Twitter, Flume, Kafka and so on. The spark engine handles all this tools to process streaming. In this paper we describe the streaming data flow from HDFS and dynamically created data. The scale programming language is coded to handle those streaming data. The streaming and static data results are stored back to the HDFS for future analysis as shown in figure 2.

V. EXPERIMENTAL APPROACH

A. WORK ENVIRONMENT

The experiment is composed on single cluster component. We use the operating system Windows 8.1 with 64 bit OS and enable the virtualization in the BIOS. Now we setup an virtual environment where Hadoop and Spark are configured in a Linux based operating system Cloud-era. This begins with Installation of latest virtual box and import Cloudera's CDH5.5 packages which provide a framework of different in-built tools. It consists of Hadoop 2.6 and spark 1.5.2 versions. Imported Scala 4.3.0-vfinal-2.11-linux and some spark libraries into eclipse to program streaming process. The Hardware requirements are 8GB RAM and 50GB Hard disk. Set up the whole package to run SparkQL, HiveQL and Streaming data process.

B. DATASET DESCRIPTION

This paper consists of a student Dataset of various years doing there *under graduations* and *post graduations* on different courses. The table consist of various attributes like collage code, ID , university ID, address, subject and so on.

The Application is built to analyze the number of UG & PG students studied in particular subject and make a year wise analysis report. This analysis may be used for many statistical report to their respective country where number of students graduate every year, literacy rate and skills of students according to their stream which reflect development in a country .

VI. APPLIED EXECUTION

A. BATCH QUERIES

Consider a student dataset in the form of CSV file. To perform batch process save files in the local file system. SparkQL is required to do batch process on the dataset. Inject different query based operations for the required results. Firstly, create and load tables by using spark query language then retrieve the table by "SELECT * FROM *table name* " where the result is stored in RDD as " Val *object name* = sqlContext.sql(" SELECT * FROM . . ."). By using RDD you can call the table and perform various actions on it like *"object name. GROUPBY("subject") .count().show()"* where it shows the count of subjects that students have studied.

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

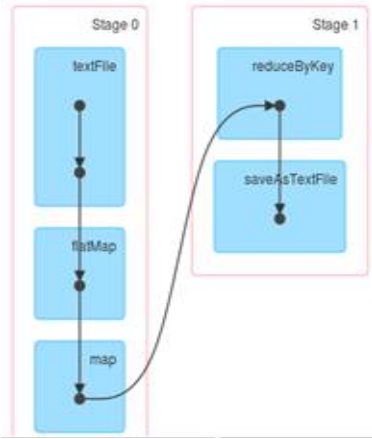
International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

B. INTERACTIVE QUERIES

Interactive operations are performed on text file where different actions are transformed to produce the result. Consider the file from local file system Save the text file to hdfs using RDD "*Val rdd1=sc.textFile("hdfs://...")*". Now flat map the text file and reduce into <key, value> pair and store the result in RDD. "*val rdd2 = rdd . flatMap(line=>line.split("")).map(word=>(word,1).reduceByKey ((a,b)=>(a+b))"* . The result of word count is retrieved using *rdd2.show()* operation. Here is the DAG visualization of word count operation using query language.



Iterative programming operations are performed on it by using RDD. As we considered it before *rdd2* object stores the result of operation performed. Now we can do various actions on it using that object like *rrd2.first(), rdd2.count(), rdd2. collect(), rdd2.filter(".. ")* and so on. As known, output of the result is stored in object, now the output will be the input for the other operation. Apache Spark Framework is best for the interactive data processing. Figure 3, shows the DAG visualization which represents the iterative process.

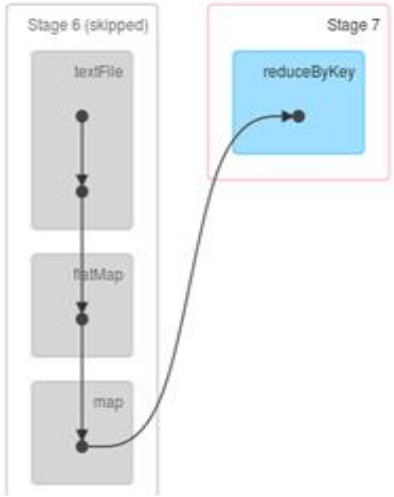


Figure 3. DAG visualization

Moving towards streaming data analysis, a Scala program is written to access the real-time streaming data. Two kinds of streaming data is analysed 1) streaming dynamically created data 2) Streaming data from HDFS and store back the result to hdfs. A word count program is performed by connecting to the local server and streaming data time is set to 10sec, where we can create data dynamically. *Val s=new StreamingContext(sc, Sec(10))*

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

Val s1=s.socketTextStream("localhost"..)
Val words=s1.flatMap(...)
words.print().
Now, Word count program is described while loading data into hdfs
Val s=new StreamingContext(sc, Sec(10))
Val s1=s.textFileStream("hdfs://....")
Val words = s1.flatMap(_.split(" "))
words.saveAsTextFiles("hdfs://....")
words.print()
The above code describes the word count program during streaming process.

VII. CONDUCT ANALYSIS

As mentioned, UG and PG student dataset is analysed as per the subjects year wise. Figure 4,5 describes the data analytics and computer science subject wise analysis of *ug* and *pg* students during their specifed years. The report is generated on the bases of dataset taken,it shows the constant result of pg students where has the ug students literacy rate has been increased or decreased year wise. Due to this survey report, steps can taken towards raise in literacy rate on there respective filelds.

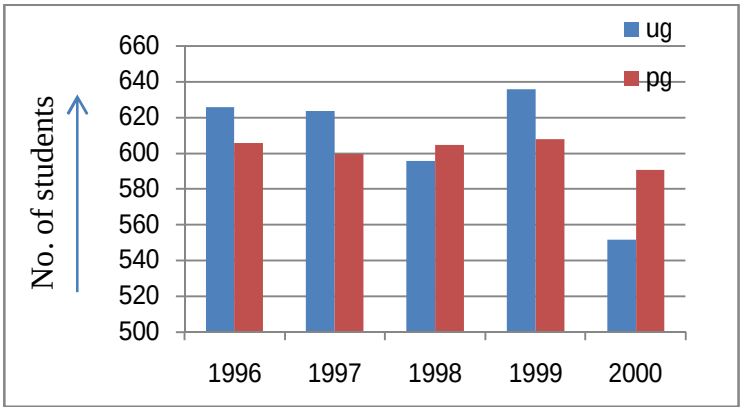


Figure 4. Data Analytics

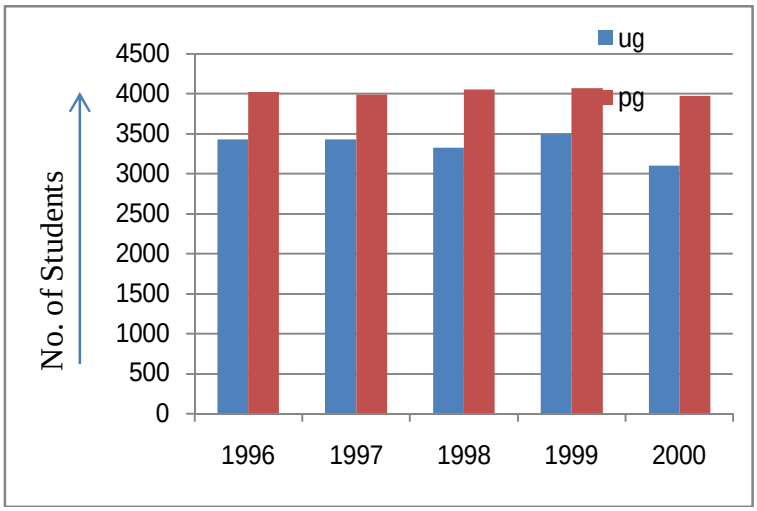


Figure 5: Computer Sciences

ISSN(Online) : 2319-8753
ISSN (Print) : 2347-6710

International Journal of Innovative Research in Science, Engineering and Technology

(An ISO 3297: 2007 Certified Organization)

Vol. 5, Issue 10, October 2016

Figure 5, shows the performance analysis of hadoop and spark. We performed multiple join operation on the student dataset using SparkSQL and HiveQL parallely to retrive the data. The report shows that SparkSQL has low latency rate than HiveQL, where hive requires more time to compute operations.

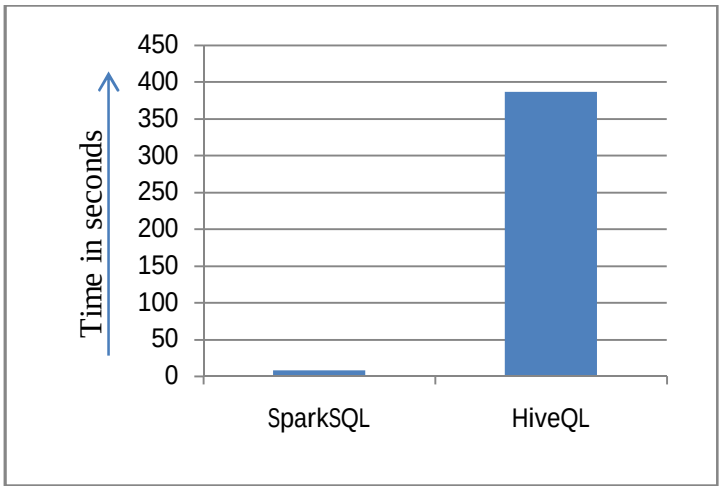


Figure 6: Latency of hadoop and spark

VIII. END NOTE

BigData has become a trend and some solutions have been provided, among them Apache Hadoop is well-known. Hadoop is designed for batch and high throughput job execution and it is suitable for jobs that process large volumes of data in a long time. However hadoop can process streaming data but further it can more efficient. Due to high demand for interactive queries and big data streams, in-memory shines as notable solution that handle both real-time and streaming data requirements. So we discussed an frame work Apache Spark which is the good example for this cases supporting in-memory computing using RDD's. And also it shows the performance analysis between Hadoop and Spark.

REFERENCES

[1] Altti Ilari Maarala, Mika Rautiainen, Miikka Salmi, Susanna Pirttikanges and Jukka Riekki "Low latency analytics for streaming traffic data with Apache Spark" proceedings of the IEEE-2015, doi:10.1109/ BigData. 2015 . 7364101
[2] "Spark: A Big Data Processing Platform Based on Memory Computing" by Zhijie Han, Yujie Zhang, 2015-IEEE , DOI: 10.1109 /P AAP .2015.41
[3] Lei Gu, Huan Li, "Memory or Time:Performance Evaluation for Iterative Operation on Hadoop and Spark ", High Performance Computing and Communications & 2013 IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC), 2013 IEEE 10th International Conference on Date of Conference:13-15 Nov. 2013Page(s):721 – 727.
[4] "Performance comparision of hive, impala and spark SQL" by Xiaopeng Li, Wenli Zhou, doi:10.11109/IHMSC.2015.95
[5] Satish Gopalani, Rohan Arora "Comparing Apache Spark and Map Reduce with performance analysis using K-Means" international Journal 2015
[6] "Streaming Twitter data analysis using spark for effective job search" Lekha R.Nair , Sujala D. Shetty , Journal of Theoretical and applied information technology - 2015
[7] Xinyi Liao, Zhiwei Gao,Weixing Ji, Yizhuo Wang "An Enforcement of Real Time Scheduling in Spark Streaming" doi:10.1109/IGCC.2015.7393730
[8] M. Zaharia, M. Chowdhury, S. S. Michael J. Franklin, and I. Stoica, "Spark: Cluster computing with working sets," *In HotCloud*, June 2010.
[9] "Lambda Architecture for cost-effective Batch and speed big data processing" by Mariam Kiran, Peter Murphy, Inder Monga, Jon Dugan, 2015-IEEE, doi:10.1109/BigData.2015.7364082
[10] "Performance Prediction for Apache Spark Platform" by Kewen Wang, Mohammad Maifi Hasan khan @ 2015 IEEE 17th international conference on HPCC, DOI:10.11.1109/HPCC-CSS-ICESS.2015.246
[11] M. Zaharia, M. Chowdhury, T. Das, et al. "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing", Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. 2012, pp. 2-2.
[12] "An optimal approach for social data analysis in big data" Kamala.V.R, L.MaryGladence,2015-IEEE, doi: 10.1109/ICCPEIC.2015.7259464
[13] Taoying Liu,Jing Liu,Hong Liu,Wei Li,"A performance evaluation of Hive for scientific data management", Big Data, 2013 IEEE International Conference on DOI: 10.1109/BigData.2013.6691696 Publication Year: 2013 , Page(s): 39 – 46.