Storing and Handling Complex Content for Large-scale Data

Hong Li Xu, Hong Hua Jiang, Qiu Lan Wu, and Yuan Yuan Wang
College of Information Science and Engineering, Shan Dong Agricultural University, Shandong Taian 271018, China
Email: coco2006@sdau.edu.cn; j_honghua@sdau.edu.cn; zxylsg@sdau.edu.cn; wyy@sdau.edu.cn

Abstract —At present, the increasing growth and pervasive development of mass data raise the challenge for big data storage. When it came to large-scale data, storage and handing method of complex data were the key problems to be settled. From big data source, we studied several problems of massive data: the big data platform framework and landing, the storage mechanism and data handling of large-scale data, the node-fault handling of mapreduce and pregel superstep, spark streaming based on the storage memory, and BlueDBM-The other forms of memory storage in massive data etc. It offered storing and handling complex content for large-scale data and support the further research of big data technical challenges, support for the application of it.

Index Terms—big data platform; node fault-tolerant; streaming data handing; BlueDBM

I. Introduction

The increasing growth and pervasive development of mass data which produced by Internet, Cloud technologies, other social networks are enabling the interconnection of heterogeneous information systems involved with “big data” technologies. “Big data" is considered to be the new "oil" in the information age whose technologies represented by Hadoop and spark ecological circle and cloud technologies [1] [2]. Its characteristics are not only the huge volume of data, but also its diversity, heterogeneous and autonomous. Accompanied by the accelerated growth of the digital universe, different data sources churn out heterogeneous data type which can be stored in RDBMS in a structured format or produced by connected devices and social networks in a heterogeneous format such as noSQL format, geographic Data, etc.[3] When it came to large-scale data, storage and handing method of complex data were the key problems to be settled. In this paper, we focused on several critical problems during the course of big data processing, including The big data source and classification, big data storage, the handling of node failure, fault-tolerant mechanism, new form of memory storage. The load balance of flow calculation etc, which highlight the storing and handling of massive data.

Manuscript received May 28, 2018; revised November 28, 2018.
This work was supported by ShanDong agricultural university youth fund .under Grant No. 24053
Corresponding author email: Coco2006@sdau.edu.cn
doi:10.12720/jcm.13.12.763-768

A. The Data Source

Web log data, which mainly came from the enterprises, was the main big data source. The main format of it was the log files [4], [5]. In addition, the production data of offline mainly reflected on the existing documents and textual data. Big data source include data over internet and smart phones. The internet data, the geographic information system data and others, they extended the boundaries of big data. Table I elaborately discussed the data classification and characteristics.

II. Big Data Platform, Framework and Landing

It need to ground storage for the massive data to do further processing.In general, there are many open source scheme of distributed file system, such as typical HDFS, but more is in the data warehouse, or in noSQL. Which represented by the hive, hbase,oracle, in Hadoop ecosystem, as shown in Fig. 1.

The Hadoop ecosystem mainly involve the following: MapReduce is a programming model for large-scale data set of parallel computing. It includes two faction Map (Map) and Reduce (reduction), the idea of it is 'divide-and-rule'. First distributed tasks to multiple nodes, with cluster parallel computing, and combine the calculation so as to achieve the final results. There exist multi-node computing, task scheduling, load balancing, fault tolerance, which are performed by mapReduce framework.

There is a commonly used metaphor for big data: the big data ecosystem is similar to the kitchen tools. Different cooking such as Chinese food, French food, Japanese food, each need different kitchen tools. And every tool has different characteristics, suitable for large data of different stages.

In view of the embodiment of the big datg value, for part professional work, the processed data are can be used directly, for example, through the operation of the data warehouse, direct external display; And some datas are need to be statistical analysis, such as the various operations through the Hive. They put the script and SQL language into program, throw it to the calculation engine, and you will free from tedious program. Products kinds of BI report, from them find the rule of the existing data, and support the decision making;

The other massive datasets tool are Mahout, and Spark MLlib etc. [6]; Graph calculation systems were also used

in massive datasets mining, it is mainly used to analysis the similarity and the link between massive data graphs.For example GraphLab, it classifies traditional machine learning algorithms into three levels, assemble, application and spread, its graphs computational efficiency is superior to MapReduce [7], [8], in addition to the above mentioned google Pregel etc. [9]. Another part of the business is to put these data as the search data sources, to improve the query speed, at this point index the data movement was required to be done.

TABLE I: DATA CLASSIFICATION

Data classification	structure	data size	describe	example
Master data	structured	low	Enterprise data entity, which is a typical non transactional data, and closely related to the strategic value of the enterprise	Customers, products, agents, etc.
Transaction data	Structured and semi-structured	Medium-high	Business transactions which born in business operation	Purchasing records, browse, query and payment data,etc.
Reference data	Structured and semi-structured	Low-medium	The internal management and external source to effectively support the handle affairs of enterprise, manage master data and provide decision support.	Geospatial data and market data
Analysis data	structured	Medium-high	for business operations and meet the demand of report .	the warehouse data, data market and other data in the decision support application
Metadata	structured	low	The concept is the "data about data"	The data name, data dimension or unit, data entity definition
document and content	Non-structured	Medium-high	Documents, digital images, geographical spatial data, and multimedia files	Claim form, medical images, and video files, map, etc
massive data	Structured, semi-structured, and unstructured	high	the challenging large data sets used in storage, search, share, visualization and analysis.	The collection of above

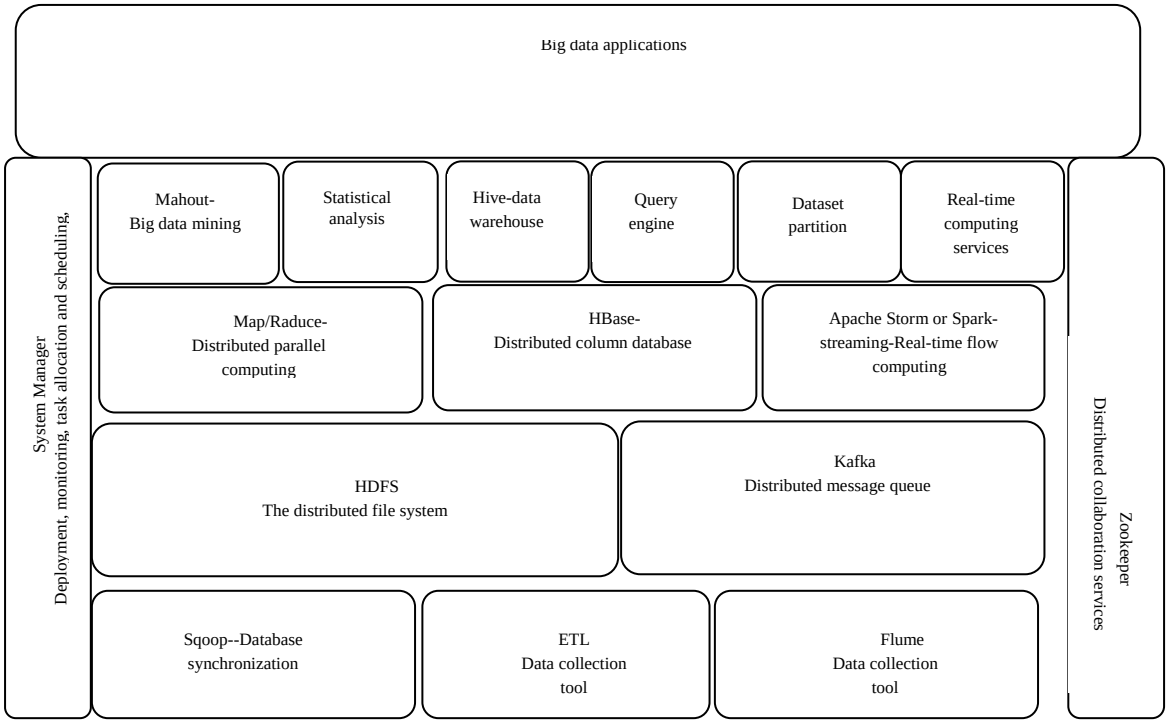

Fig. 1. Big data technology roadmap

III. THE STORAGE MECHANISM AND DATA HANDLING OF LARGE-SCALE SCALE DATA

The requirements for large data storage are as following:

- High availability, which requires the data can be accessed at any time.
- Low cost, which demand disk capacity.
- High performance, it means the access velocity is fast.
- Low overhead, it means the less usage of CPU and network resource.
- The key technologies for high availability: multi-copy technology and disaster tolerance technology.

Among them, HDFS, GFS, MooseFS, etc is on behalf of multiple copies technology. When a copy fails, the system automatically redistributes the data and returns multiple copies as soon as possible (usually one copy).

Multiple copies bring not only usability benefits, but also performance benefits.

GFS(Google File System) is an extensible distributed file system for large, distributed applications that access large amounts of data. It runs on cheap, generic hardware and provides fault-tolerant functionality. It can provide a large number of users with high overall performance.

A GFS cluster consists of a master and a large number of chunkservers, as is shown in Fig. 2, and it is accessed by many clients. The master and chunkserver are usually Linux machines which running the user layer service process. Chunkserver and client can run on the same machine as long as resources and reliability are allowed. The file is divided into blocks of fixed size. Each block is identified by a constant, globally unique 64-bit chunk-handle that is assigned by the master at the time the block created. ChunkServer stores blocks as Linux files on a local disk, it can read and write data specified by chunk-handle and byte range. For reliability, each block is copied to multiple chunkservers. In default, three copies are saved, but this can be specified by the user.

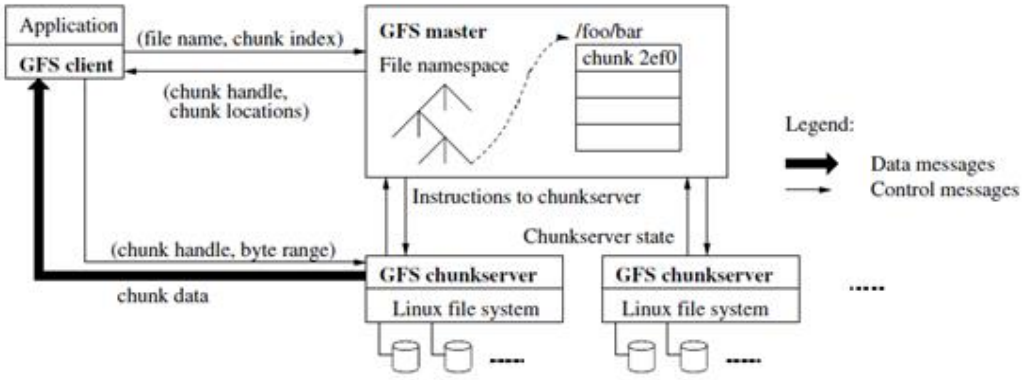

Fig. 2. GFS architecture

A. HDFS System

The second is HDFS system. The aforementioned challenges are strongly contribute to big data storage whose system often adopt distributed file system (DFS), the distributed file system is the prototype of the google file system, which is presently used HDFS (Hadoop Distributed File System), their common characteristic is: the file is big, even reached TB level, and less file update, usually take files as the input data, and there is often an additional data appended to the file's tail. Therefore, for frequently updated data system, it is not suitable for DFS.

In DFS, the files is divided into many blocks (chuck), file chuck will be copied to different duplicates on a few computing nodes (e.g. 3), these copies are not stored in the same rack in case of a rack recovery after system failure. In order to search one file, there are file directories and master node file, the purpose of the former was to look for the master node duplicates, and the master node attended to search a file chuck. The total directory and the master node have several copies too, the user comprehend the total directory location.

In terms of data consistency, GFS is better than HDFS in theoretically:

a) GFS provides a relatively loose consistency model. It supports both write and record additional operations, the write operation allows us write file randomly, and the record appending supports parallel operation more safe and reliable.

b) HDFS has the same function as GFS for writing operations, but it does not support record appending and parallel writing operations. NameNode uses INodeFileUnderConstruction attribute to mark the block of the file that is being manipulated, rather than paying attention to the read or write operation. Even the DataNode can't see the lease. Once a file is created, written, and closed, there is no need to modify it. This simple model is suitable for Map/Reduce programming.

B. The Node-fault Handing of MapReduce and Pregel Superstep

When it comes to the node-fault of MapReduce, it including the collapse of a running map node and other reduce node failures, for the former, the master control program which regular inspect the collapse of all nodes, will find the node collapse. As a result, all originally assigned tasks of the default map node will have to be perform again, sometimes need to restart the completed map tasks. For the latter, when it run into an reduce node-fault, the master process will make the node free and reschedule the work node to complete the task. With regard to node-fault, many expanded version of MapReduce take it into account. They provide most of the task execution process in the node-fault treatment, so as not to restart the whole system.

C. The Storage of Regular Data

As far as the structured information, such as two - side tables which usually utilized in RDBMS. There storage platform often transfer from RDBMS to big data. At this circumstances, the projects regularly choose Oracle as their database because of its appropriate storage construction represented by its optimized physical ,logic and memory structures ,based on it ,projects can reach its

better performance in reading and writing .And project selected Sqoop as the transfer tool. Project need to specify the connection string, the table name and field delimiter, - num- or -m- mappers parameter utilized to perform the mapping task (parallel process) number which import data from Oracle. As exhibited in following (In this case, we only use a mapper. (rac1)):

```
[wissem@localhost ~]$ sqoop import
--connect
jdbc:oracle:thin:@//sandbox1.xxx.com:1522/orawissto
HDFS
--table USER_ACTIVITY  --fields-terminated-by
'\t'
--num-mappers 1 --username WISSEM -P
```

Sqoop automatically create a jar file, called "USER_ACTIVITY. Jar" and use sql to extract Oracle database table User_Activity contents. The HDFS file named part-m-00000. This is FileOutputFormat class defined in the Hadoop file naming rules. Because project defined only one map therefore there existed one output file on big data HDFS storage platform called part-m-00000.

D. The Data Storage and Transfer of Streaming Data Information in Big Data

The unstructured data play an important role in big data universe; it contributes to NOSQL data storage with large-scale, distributed scalability feature in the case of enormous amount of unstructured data.

In Wikipedia: A NoSQL database provides a mechanism for storage and retrieval of data that is modeled other than the RDBMS [6], [7]. Three data model approaches are usually used in NoSQL: Key-Value and Wide Column Model Key-value (such as Google BigTable), document base (such as MongoDB)model, Graph Model (such as Neo4j and HyperGraphDB).

Document based storage contains several different key-value pairs, or key-array pairs, or even nested structure. Document database are work for a wide variety of applications due to the flexibility of the data model, the ability to query on any field and the natural mapping of the document data model to objects in modern programming language. For example, when it comes to the GIS (geographic information system) big data, Esri support spatial framework for Hadoop platform. The GIS Tools for Hadoop is open whose framework including the UDF of Hive and SDK of JSON. It set up advantageous bridges between the big data and GIS system.

E. Based on the Memory Storage (Spark Streaming)

The computing model of big data is mainly divided into batch computing, stream computing, interactive computing, graph computing, etc. Among them, flow calculation and batch calculation are two major data calculation models, which are applicable to different big data application scene respectively. For data calculation without first storage, the real-time requirement is very strict, but the accuracy of data is often not very demanding, and the flow calculation has obvious advantages.

1) Fault-tolerant mechanism of spark streaming

The spark streaming kernel concept is RDD (resilient distributed dataset) which is flexible, distributed, partitionable dataset with fault-tolerant mechanism, and often produced in function modes [8]. The RDD is a kind of special dataset with fault-tolerant. It provides a read-only, shared memory mechanism based on existing RDD transformation, and all the data will be loaded into memory for convenient reuse. It is distributed and elastic, can be distributed on multiple machines with flexible data processing. When it comes to insufficient memory, the system will transform data between memory and magnetic disk auto. These limits can greatly reduce the automatic fault-tolerant. It is a kind of more general iterative parallel computing framework [9], [10]. The internal processing mechanism of Spark Streaming: receive real-time Streaming data, and according to the time interval take data into streams of data batch, and then data processing through the Spark Engineer deal, finally get processed the outcome result set.

```
// through Socket method to get data,it need the
host name and socket port number,the data will be
stored in memory and hardware.
    val lines = ssc.socketTextStream(args(0),
args(1).toInt,
StorageLevel.MEMORY_AND_DISK_SER)
// write to HDF system
 rowrdd.foreachRDD(rdd =>if(rdd.count()!=0){
  val out = rdd.map(_._1).first().toString
   rdd.map(_._2).saveAsTextFile("/user/root/hive/
jboss/"+out)
ssc.start()
```

The project can use Sparking Stream to handle heterogeneous source data, such as RDBMS, HDFS, Log file, network streaming data. It need to pay attention that every batch duration of Spark Streaming can continually produce RDD. In this circumstance, the treatment model of null RSS will affect the system efficiency. The method rdd.isEmpty() returns a Boolean value. This would be especially useful when using streams. At this time, it need to judge whether their RDD exists in scheduler layer.

It will take advantage of several RDD funcitons to carry out the data transformation work, they are demonstrated in Table II, including saveAsTextFile, saveAsSequenceFile, saveAsObjectFile and so on.

2) The Load banlance of flow calculation

In the big data flow computing environment, load balance mechanism of the system is stable operation system High throughput computing fast response of a key factor. However, the current majority of load balancing system can not effectively support system, such as Storm, S4 system does not support load balancing mechanisms, such as Kafka system has realized the support for load balancing mechanism parts: on the one hand, in the big data flow computing environment, data rate of system has

obvious respectively, and often unable to effectively predict the duration, this leads to in the traditional environment has the very good effect of load balancing strategy theory and practice will no longer applicable in big data streaming computing environment[11]; On the other hand, the current big data streaming most open source computing system on the design of the architecture has yet to fully consider the load balancing problem of the whole system, in the practical application, and relatively lack of related experience, therefore, to the big data flow computing environment has brought a lot of practice of the research on the load balancing problem of the difficulties and challenges.

Big data flow of load balancing problem in the computing environment, requires a combination of specific application scenarios, systematically analyzes and summarizes the hidden in large numbers according to the data flow calculation of the basic characteristics and inherent regularities of the variation of the flow, combined with the experience of the traditional load balancing system, according to the practical situation, the ongoing mechanism of continuous optimization and gradually improve.

TABLE II. THE FUNCTIONS OF ADD TRANSFORM

Functions	The Function Contents
saveAsSequenceFile(path)	The format of the data set of elements to sequencefile, saved to the specified directory, the local system, HDFS or any other hadoop file system support. RDD elements need to be the key - value pairs, and implements the Hadoop Writable interface, or implicit can be converted into Writable (Spark includes basic types of transformation, such as Int, Double, String, etc.)
saveAsTextFile(path)	The elements of data sets, which in the form of textfile, will be saved to a local file system- HDFS or any other file system supported by Hadoop. Spark will call tostring method of each element, and convert it to a row text of the file.
saveAsObjectFile()	Turn RDD to array struscture, and do the serialization operation, the results are directly saved in hdfs.

F. BlueDBM-The other forms of Memory Storage

In big stream calculation, the data are in tuples, the continuous data flow to computing platform with multiple data source. The flow data is dynamic, multiple applications and high velocity, which can be expressed through their the task graph respectively.

Streaming data storage is the key technology. In the calculation, too much data will lead to data overflow in the case of insufficient memory. We need to do the balance between cluster and quantity of data evaluation. Storm abstracts messaging, processing flow calculation on cluster machines automatically and concurrently. However storm is not including the storage concept [12], [13]. Usually its data will be storage in hbase or hive system. Instead, S4 system provides a persist API, It will store the output according to the time or capacity.

Kafka message system implemented by means of adding data to persist the disk data, realize the stability of big data storage and effectively improve the capacity of the massive data system [14], [15]. By using Sendfile system that would call functions to optimize the network transmission, it reduced memory copy one time, as a result to increase the throughput of the system.

At present, the concept of BlueDBM is introduced into big data storage. BlueDBM is a system architecture to accelerate big Data analytics which has a large distributed flash-based storage with in-store processing capability and a low-latency high-throughput inter-controller networks. When processing massive amounts of data, performance if often bound by the capacity of fast local DRAM. In cluster systems with more RAM, the network software stack often becomes a bottleneck.

BlueDBM proposes to mitigate these issues by providing an extremely fast access to a scalable network of flash-based storage devices, and to provide a platform for application-specific hardware accelerators on the datapath on each of the storage devices. The diagram below shows the difference between in-datapath and off-datapath architectures. BlueDBM provides both architectures by wrapping the storage device under a consistent abstraction.

IV. CONCLUSION

The main context and key problem of massive data storage was studied in this paper. When it came to large-scale data, storage and handing method of complex data were the key problems. From big data source, we studied several problems of massive data: the big data platform framework and landing, the storage mechanism and data handling of large-scale data, the node-fault handling of mapreduce and pregel superstep, spark streaming based on the storage memory, and BlueDBM-the other forms of memory storage in massive data etc. It offered storing and handling complex content for large-scale data and support the further research of big data technical challenges, support for the application of it.

ACKNOWLEDGMENT

The work was supported by ShanDong agricultural university youth fund NO. 24053.

REFERENCES

[1] J. Feblowitz, "Analytics in oil and gas: The big deal about big data," in *Proc. SPE Digital Energy Conf*., 2013.

[2] M. Jorge, C. Ismael, R. Bibiano, S. Manuel, P. Mario, "A data quality in use model for big data," *Future Generation Computer Systems*, vol. 63, pp. 123-130, 2016.

[3] Y. S. Dong, D. C. Tao, X. L. Li, *et al*., "Texture classification and retrieval using shearlets and linear regression," *IEEE Trans Cybern.*, 2014.

[4] R. R. Ji, Y. Gao, W. Liu, *et al*., "When location meets social multimedia: A survey on vision-based recognition and mining for geo-social multimedia analytics," *ACM Trans Intell. Syst. Technol.*, 2014.

[5] B. Shao, H. Wang, Y. Li, "Trinity: A distributed graph engine on a memory cloud," in *Proc. Int'l. Conf. on Management of Data. ACM*, 2013, pp. 505-516.

[6] S. Wolfert, L. Ge, C. Verdouw, M. J. Bogaardt, "Big data in smart farming – A review," *Agricultural Systems*, vol. 153, pp. 69–80, 2017.

[7] X. Q. Cheng and X. L. Jin, "Survey on big data system and analytic technology," *Journal of Software*, vol. 25, no. 9, pp. 1889-1908, 2014.

[8] J. Z. Li and X. M. Liu, "An important aspect of big data," *Journal of Computer Research and Development*, vol. 50, no. 6, pp. 1147−1162, 2013.

[9] E. Ahmed, I. Yaqoob, I. A. T. Hashem, I. Khan, *et al*., "The role of big data analytics in internet of things," *Computer Networks*, 2017.

[10] G. Malewicz, M. H. Austern, A. J. C. Bik, J. C. Dehnert, I. Horn, N. Leiser, and G. Czajkowski, "Pregel: A system for large-scale graph processing," in *Proc. International Conference on Management of Data*, 2010, pp. 135–145.

[11] H. Demirkan and D. Delen, "Leveraging the capabilities of service-oriented decision support systems: Putting analytics and big data in cloud," *Decision Support Systems*, vol. 55, no. 1, pp. 412-421, 2013.

[12] D. He, N. Kumar, H. Wang, L. Wang, and K. K. R. Choo, "Applied mathematics and computation journal homepage," *Applied Mathematics and Computation*, vol. 314, pp. 31–43, 2017.

[13] F. Celesti and M. Puliafito, "Villari university of messina, C.da Di Dio - sant'agata big data storage in the cloud for smart environment," in *Proc 6th International Conferenceon Ambient Systems, Networks and Technologies*, 2015, pp. 500-506.

[14] R. Nachiappan, B. Javadi, R. N. Calheiros, and K. M. Matawie, "Cloud storage reliability for Big Data applications: A state of the art survey MARK," *Journal of Network and Computer Applications*, vol. 97, pp. 35–47, 2017.

[15] L. Lim, A. Misra, and T. L. Mo, "Adaptive data acquisition strategies for energy-efficient, smartphone-based, continuous processing of sensor streams," *Distributed and Parallel Databases*, vol. 31, no. 2, pp. 321−351, 2013.

Hong-Li Xu was born in 1976, Associate professor, Ph.D. in electrical engineering, College of Information Science and Engineering, ShanDong Agricultural University, Shandong Taian. She received the PhD .degree from School of Mechanical electronic & information engineering, China University of Mining & Technology, Beijing 100083,China,in 2014, and her doctor thesis obtained excellent doctor degree dissertation. Research direction: chaos and intelligent algorithm, big data, etc.