Internals of Hadoop Application Framework and Distributed File System

Saminath.V, Sangeetha.M.S

Abstract- Hadoop is an open-source framework to storing and processing of Big data in a distributed environment. Big data is collection of complex and large volume of structured and un-structured data. Hadoop stores data throughout clusters located in geographically different machines and distribute workload using parallel computing. MapReduce is software framework derived on Java, to analyze the large scale data. MapReduce uses Distributed Data processing model. HDFS is another component in Hadoop, storing large volume of data. Google File system supports immense amount of data stored into distributed data nodes, each node has redundant data storage maintained to avoid lost. This paper explains the HDFS, details of jobs node cluster environment, stack layered component on Hadoop framework, various Application development on Hadoop.

Index Terms- HBASE: Big Data Base, HDFS: Hadoop Distributed File System, JDBC: Java Data Base Connectivity, MapReduce, Hive, Pig, Sqoop, SQL: Structured Query Language.

I. INTRODUCTION

Big data can be classified into '3V': Variety, Velocity and Volume. It increases storage capacity of data, increasing adequate processing power. Data can be stored in volumes like Zettabyte/Petabytes, forms of data can be structured or unstructured, and data can be processed in Batch sequence process or streaming and parallel processing. Data inputs from several Sources like, Data generated from IOT connected service, Social media generated data from emails, tweets, photos, click stream by advertising, and appliance generated data from various sensors, logs and other information.

Data model is categorized as Structured, Semi-Structured and Un-Structured types stored in cluster system. Figure-1 explains Hadoop Stack structure information around big data components. Hadoop Core framework provides effective analysis on Data Models. MapReduce development based java programming, it access the Hadoop core framework and HDFS is distributed file system with file manipulation implementation. Both MapReduce and HDFS is low level component, directly access the Hadoop Core framework. Layer-2 component are access the Low level component and retrieve the results from Hadoop core component. PigLaltin, HiveQL and Hbase, Sqoop and other open source components are supports to communicate with Low level component. High level component has interpreters, convert to Low level component and to execute jobs on the cluster. High Level component is developer friendly framework, easy to analysis content in BigData cluster. PigLatin is developed based on Scripting language, HiveQL based on SQL Query Language.

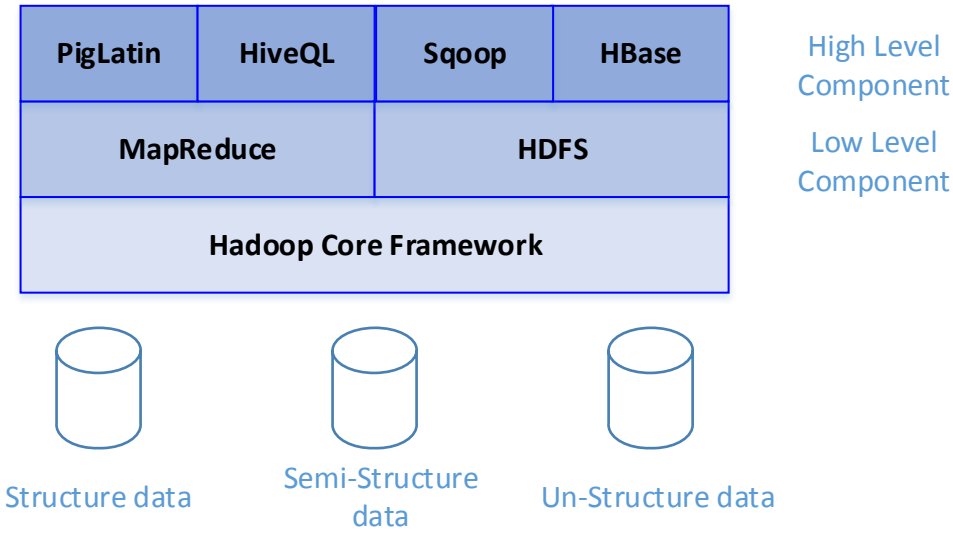

Figure-1: Hadoop Stack Framework

MapReduce component is developed on Java programming, It collects and analysis data on various domain like, Financial service for Banking, Heath care service for Medical system, Telecom domain, Energy domain, Travel and Logistics Domain. Data Analysis to Image processing, Intelligent Business tool development, Data Analytical tool, monitoring data streaming.

II. HDFS (HADOOP DISTRIBUTED FILE SYSTEM)

HDFS is derived from Google-File-System (GFS), completely developed on Java platform and runs over the Linux File system Ext3 and Ext4. Data spirited into blocks and stored in Cluster devices. Each block of data replicated and stored into three minimum distributed location. If any failure notification on the data, HDFS automatically load missing data from other cluster. Figure-2 illustrated Hadoop File system structure and node details.

HDFS data stored into two nodes

1) NameNode (NN) : Act as Master, Stores Metadata information
2) DataNode (DN): Act as Worker, Stores Actual Data block information.

Name Node stores Metadata information about actual data location of each data block, file name and file properties. Name Node content loaded into RAM, to access quickly. Size of Name Node is restricted by the File system because constrain in RAM memory. Data Node has actual vast data stored into blocks. Data needs to be stored into continuous blocks of node. Data are replicated across several distribute location and receive reports from each location. Name Node and Data Node are synchronized, Data Node send notification to Name Node. If fails, Name Node assumes that Data Node is lost. Name Node find the blocks of Data Node lost from its meta information, copy the data block from other nodes. Heartbeat communication between Name Node and Data Node is maintained. All Data Node connected in pipe-line architecture. New Data Node loaded from remote location, previous and Next Data Node linked to new Data Block. Secondary Name Node keeps update Meta data information and periodically compare with main Name Node information.

User level libraries are available to access the HDFS. Hadoop command line interface to remove/add directory Copy data to/from HDFS, Display content from HDFS, validate permission of HDFS information. Java API also provides framework to access the HDFS data structure. All the Jobs (Mapper, Reducer) runs on the Client machine and access the Data in File system using HDFS protocol.

Characteristic of HDFS

1. Write once, cannot be updated.
2. No Random write files allowed.
3. Random Reads with multiple times

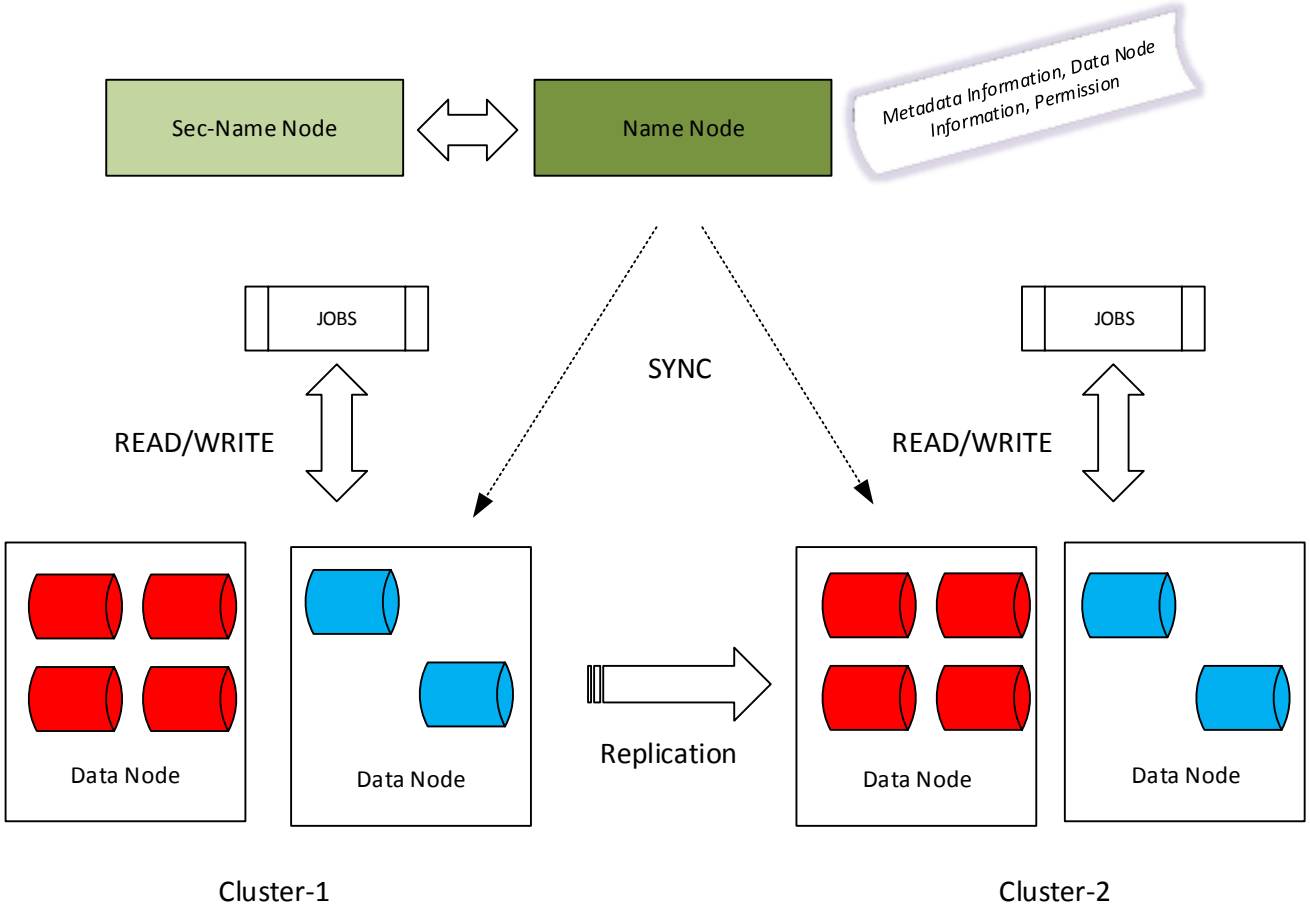

Figure-2: HDFS Architecture

III. HADOOP COMPONENTS

Hadoop is evolving framework in big data analysis, lots of component are available to access the data for different purpose. Some of the components can be analysis below.

A. Mapper and Reducer Architecture

MapReduce is java based software framework, to process large amount of data in Distributed cluster. MapReduce framework consist of single master JobTracker and one slave TaskTracker in each cluster. Master JobTracker schedule jobs, monitor all the jobs, if required reschedule jobs. Hadoop Job client is jar executable, it configures JobTracker and responsible for execute and configure slave TaskTracker application. Mapper task takes input key/value pair from HDFS. Mapper process the input data (key, value) convert to intermediate (key, value) records. Mapper reads input by specifying an InputFormat, currently system supports InputSplit and RecordReader formats. Text data processed by TextInputFormat and TextOutputFormat, each line of text data read and write to HDFS. Same way read and write in SequenceFile using SequenceFileInputFormat and SequenceFileOutputFormat functions.

The Reducer has a reduce method that transforms intermediate data (key, value) aggregates into smaller set of output (key, value) pairs. Reducer has 3 prime implementations are Shuffle, Sort and Reduce. The shuffle and sort framework runs simultaneously, it fetches mapper outputs and processed it. Reporter application use to generate status information and counter details write into file, which identify the live information about task and process. Blocks of M-R component explained in Figure-3.

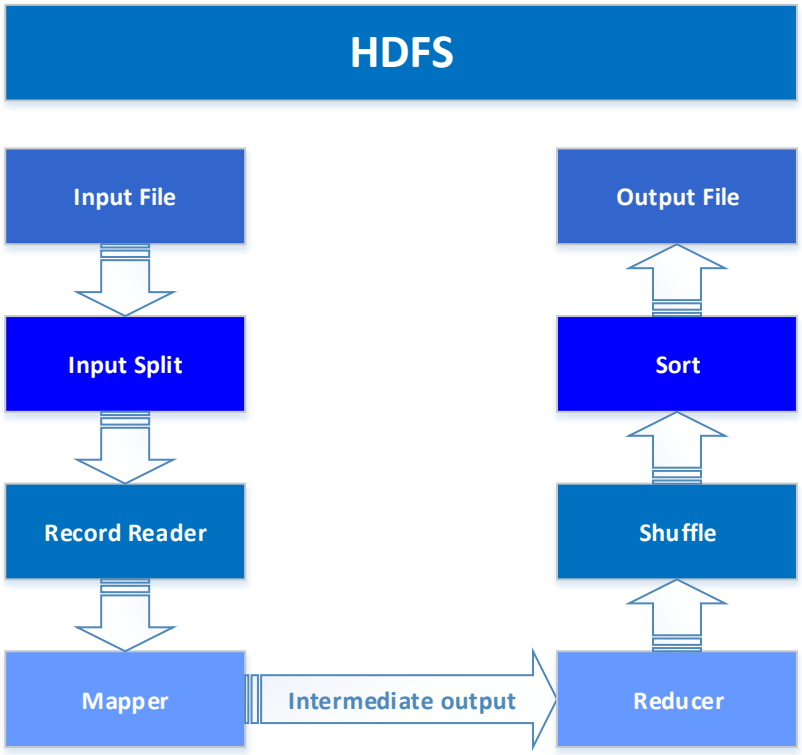

Figure-3: Mapper-Reducer Streaming data flow

B. HiveQL

HiveQL is SQL Based Query Application to supports MapReducejobs and executed across a Hadoop cluster. Hive allows SQL developers to write Hive Query Language (HQL) statements that are similar to standard SQL statements. Map reduce code is typically written in java and requires a developer with a good understanding of Java, Map reduce paradigms, Hadoop API's etc. But Hive is language based on SQL, Easy to learn, easy to use, immediately makes data on the cluster accessible to many more people and allow non-java programmers access to the dada in its Hadoop clusters.

Hive Implementation:

1) The Hive takes interpreter runs on a user's machine
2) Takes Hive queries and turns them into java Map reduce code
3) Submits the code to the cluster
4) Displays the result back to the cluster

C. PigLatin

Pig is a data flow language composed of series of statements (sequence of operations or transformations) with relatively simple syntax. Pig scripts are turned into Map Reduce jobs and executed on Hadoop cluster like HiveQL. The Pig interpreter runs on the client machine, it converts Pig Latin sequential script to standard Java Map reduce jobs, which are then submitted to the job tracker. Pig is originally created by Yahoo. No Modification requires on cluster, to install Pig. The PigLatin scripts translate into standard Java Map reduce jobs, which are then submitted as job tracker in client system.

Element of pig:

1) A single elements of data is an atom – Analogues to field
2) A collection of atoms is a tuple-Analogous to a row, or a partial row, in database terminology
3) A collection of tuple is a Bag-Sometimes known as a relation or result set

A Pig Latin script starts by loading one or more datasets into bags in RAM memory and creates new bags by modifying the content of loaded data structure.

D. Sqoop

Sqoop is (SQL-to-Hadoop) big data tool to extract structured data from RDBMS database and convert Hadoop supported format, also stored into HDFS data base. Sqoop supports command line interpretation to access RDBMS and HDFS database. Commands are executed in sequentially. Bulk import type import complete table/database from RDBMS to HDFS file system. Direct Input type import SQL database to Hive and HBase. Data Export type export HDFS file system to Relational Data base. Sqoop is a tool designed to transfer data between Hadoop and relational database servers. It is used to import data from relational databases like MySQL, Oracle to Hadoop HDFS and export from HDFS to RDBMS. Sqoop uses JDBC to connect with RDBMS and read/write the information. Based on the table's schema generates the necessary java classes to import sql data into the Hadoop Distributed File System (HDFS). Sqoop creates and launches a MapReduce job to read tables from the database and store to HDFS. Sqoop import tables into Hive for processing and export tabular data from HDFS to RDBMS. Communication blocks between HDFS and RDBMS using Sqoop is explained in Figure-4.

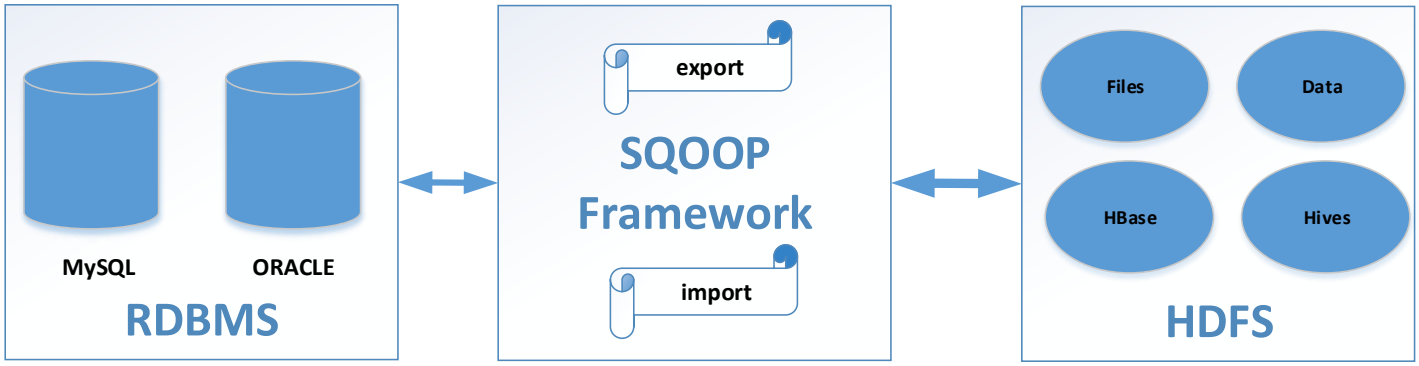

Figure-4: Sqoop Architecture

E. HBase

HDFS does not support random read/write to their file system, it need additional framework for access to sustenance it. HBase is a data model that is similar to Google's big table designed to provide quick random access to huge amounts of structured data. It leverages the fault tolerance provided by the Hadoop File System (HDFS).It is a part of the Hadoop ecosystem that provides random real-time read/write access to data in the Hadoop File System. Data can be store the data in HDFS either directly using Query Language or through HBase. Data consumer reads/accesses the data in HDFS randomly using HBase. HBase resides top of the Hadoop File System and provides read and write access to high level framework. Figure-5 describes the Hbase block between Random calls in user client and HDFS. Table-1 gives difference between Hbase and HDFS structure.

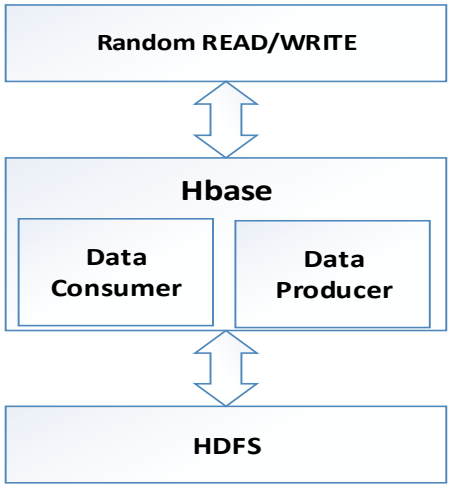

Figure-5: HBase Architecture

Table-1: Difference between HDFS and HBase

HDFS	HBASE

HDFS is a distributed file system suitable for storing large files.	HBase is a database built on top of the HDFS.
HDFS does not support fast individual record lookups.	HBase provides fast lookups for larger tables.
It provides high latency batch processing; no concept of batch processing.	It provides low latency access to single rows from billions of records (Random access).
It provides only sequential access of data.	HBase internally uses Hash tables and provides random access, and it stores the data in indexed HDFS files for faster lookups.

IV. Application Programing on Hadoop

Developing Hadoop Application by MapReduced using Java or Pig using PigLatin scripts or HiveQL using Query language. For all development we need Driver application to load target Analysis Application. Figure-6 explains sequence of data processing between Mapper and Reduce model, Input/output key/value pair details with examples. Mapper Application read data and produce Intermediate results, Reducer produces final results from intermediate output. Both Mapper and Reducer objects are loaded and controller by driver application. Source code driver, mapper, reducer are explained below tables.

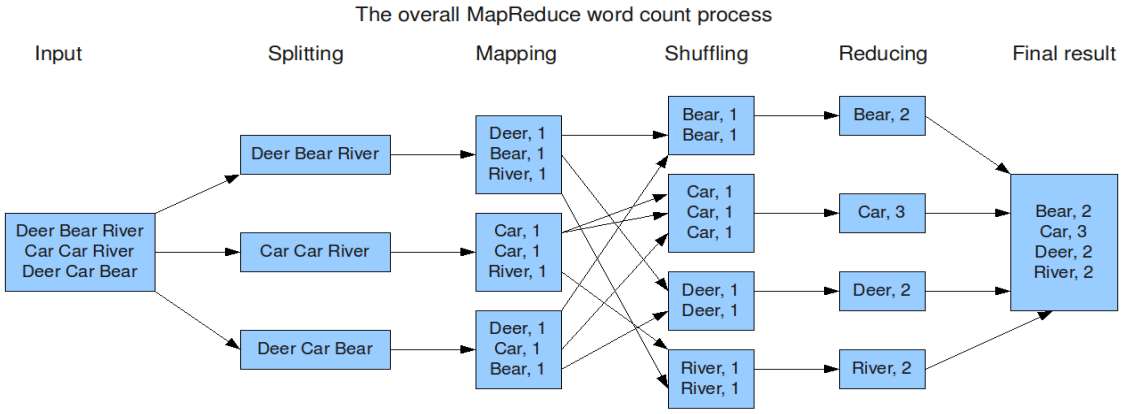

Figure-6: MapReduce Architecture

Driver Code loads Mapper and Reducer objects and run the jobs. Configure the File input and output formats and Path information's.

```
Job jobobj = new Job();
jobobj.setJobName("Word Count");

FileInputFormat.setInputPaths(jobobj , new Path(args[0]));
FileOutputFormat.setOutputPath(jobobj , new Path(args[1]));

jobobj.setMapperClass(Mapper.class);
jobobj.setReducerClass(Reducer.class);

jobobj.setMapOutputKeyClass(Text.class);
jobobj.setMapOutputValueClass(IntWritable.class);

JobClient.runJob(jobobj);
```

Mapper code reads content from File using BufferReader class, content of each word tokenized and write into intermediate context

```
public void mapper (LongWritable key, Text value, Context context) throws IOException, InterruptedException {

BufferedReader fis = new BufferedReader(new FileReader("InputFile.txt"));
while ((Line = fis.readLine()) != null) {
        StringTokenizer tokenizer = new StringTokenizer(Line);
```

```
            while (tokenizer.hasMoreTokens()) {
                        word.set(tokenizer.nextToken());
                        context.write(new Text(word), new IntWritable(1));
            }
      }
}
```

Reducer code reads intermediate output from Mapper object and collect the key with counter values. Final outputs write into context

```
public void reducer(Text key, Iterable<IntWritable> values, Context context) throws IOException, InterruptedException {

while (values.hasNext()) {
  sum += values.next().get();
}
context.write(key, new IntWritable(sum));
```

V. CONCLUSION

This paper explains the overview of Hadoop framework architecture with stacks of component. Hadoop is emerging technology in giant data analysis and big data storage planning with different structural format. Hadoop supports distributing technology to evade of data lost and system becomes stable. HDFS supports giant size of file stored in a place with replicated information in multiple geo graphical location. Hadoop core component and base component has some restriction in development like Java and not supporting of random access. The High level programming overcome to easy usage of Hadoop technology like Sqoop, HiveQL and Pig. Development point of view High level application runs top of the low level framework like Map-Reduce with efficient fashion.

REFERENCES

[1] Cloudera , "Programming with Hadoop", 2009.
[2] Milind Bhandarkar, *Data Intensive computing with Hadoop*. Yahoo.Sep/2008
[3] Serge Blazhievsky, *Introduction to Hadoop,Map Reduce and HDFS for Big Data Application*. SNIA Education, 2013.
[4] Marker Kerzner, "*Hadoop unlimited*" ,2013.
[5] Apache Software foundation, "*Map Reduced Tutorial*," 2008.
[6] Apache Hadoop "*Cloudera Developer Training*",2013

AUTHORS

First Author – Saminath, M.E. Drives and Embedded Controls, Accenture and s.x.veerapandian@accenture.com.
Second Author – Sangeetha, B.Tech Information Technology, Alethea Communications Technologies and sangeethams_84@yahoo.co.in.