Heterogeneous Data Processing using Hadoop and Java Map/Reduce

Jasmeet Singh Puaar
Student
Guru Tegh Bahadur Institute of Technology
GGSIPU, Delhi.

Ramanjeet Kaur
Student
Guru Tegh Bahadur Institute of Technology
GGSIPU, Delhi.

ABSTRACT
In this paper, the objective is to do analysis of New York Stock Exchange's heterogeneous sample data using java map-reduce on Hadoop platform. Java programming as well as Java map-reduce API has been used to work upon huge amount of data i.e. BIG DATA. The source data is of heterogeneous type. The format and the structure of the data files worked with are different.

So, it was challenging to handle the data and send it to the mappers to get a single reduced output file. The analysis of NYSE's data was done to find out the maximum and minimum price of every particular stock exchange for each year and to calculate average stock price of any stock exchange for a particular year by using record of its dividends in the sample data. This has been done by usage of data from two different files namely: dividends.csv and sample_prices.csv .The output of the program was saved to the HDFS file system. This output can then be saved to our NTFS file system using Sqoop or the files can be manually copied to our system for further processing.

General Terms
Data processing, Hadoop, Java Map/Reduce.

Keywords
Heterogeneous data processing, MapReduce, Big data, Data Analysis, HDFS, multiple input, NYSE data.

1. INTRODUCTION
Big data is undoubtedly big, but it is also a bit misnamed. It is a broad term for data that does not fit the usual buckets. Big data refers to data that is too big to fit on a single custom cluster (server), too unstructured to fit into a row-column or structured database, or too progressively flowing to fit into a constant data warehouse. While its size is the most important part, the most troublesome aspect of big data really comprises its lack of structure.[1] Big data is used when the usual application of current technology does not enable users to obtain timely, cost-effective, and efficient answers to data-driven problems. The above statement is defined by The University of California, Berkeley. [2]

Traditional data analytic techniques which works on structured data of low volume found to be inefficient to handle the variety and complexity offered by the big data which is mostly unstructured or semi-structured and that comes in huge volume. A comparison of Big Data analytics with traditional analytics is given in table 1. [3]

Table 1. Traditional Data Analytics vs. Big Data Analytics

	Traditional Analytics	**Big Data Analytics**
Data Sources	Homogenous sources providing only structured and constant data	Heterogeneous sources providing both structured, unstructured/ semi structured and streaming data
Data Storage	Isolated proprietary servers	Cloud Hosting in Public/Private/ Hybrid Cloud
Database Technology	Relational data stores (row-column data stores)	NoSQL data stores (unstructured)
Data Processing	Centralized Architecture	Distributed Architecture
Analytics	On previously collected data (static data)	Need for real time analytics (streaming data)

2. RELATED WORK
The IT industry has developed in the analysis of mainly structured data because of RDBMS being the recommended method for storing, processing and analyzing structured data. Unstructured data — everything from email, images and web logs to social media posts and sensor data — is growing at an unprecedented pace thus ignoring unstructured data is inadvisable, as effective analytics plays a vital role as a business differentiator Technologies such as Hadoop are being used universally to store and process unstructured data for analysis. [4] MapReduce, being the heart of Hadoop, is the programming paradigm that allows distributed data processing and querying by extracting data from big datasets hosted on servers in any typical Hadoop implementation. The nature of data in structured and unstructured formats being different results in different kind of analysis over the data is also different. Frameworks like MapReduce can excerpt the inmost intelligence from the entire data of the organization with the help of big data analytics capable of sourcing data from any big data sources.

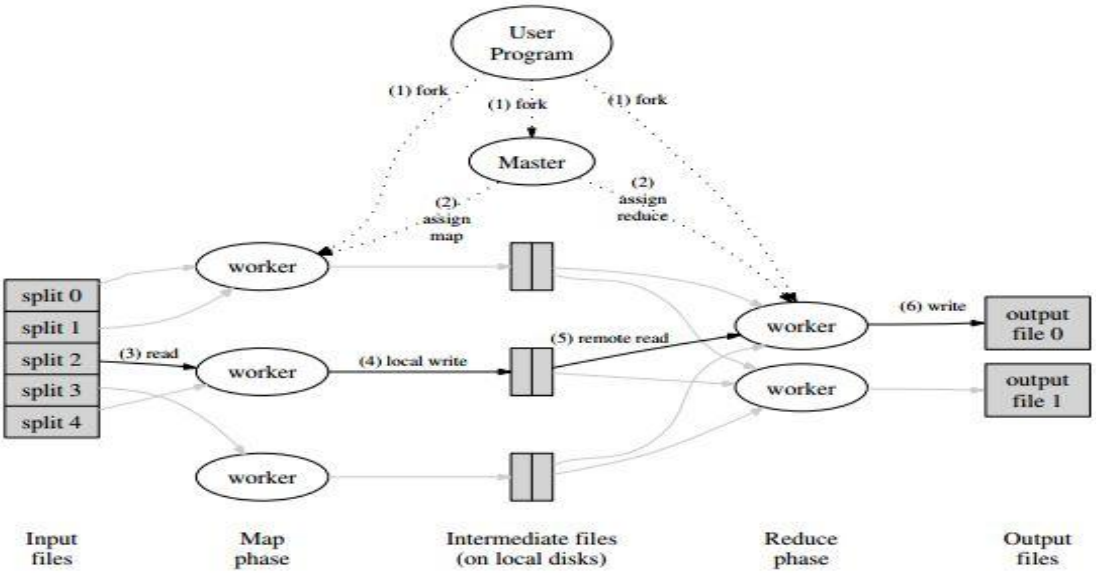

Fig 1: MapReduce Execution overview.

3. THEORETICAL ANALYSIS

3.1 MapReduce

MapReduce is a programming model and an associated implementation for processing and generating large data sets. Users specify a map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key. Many real world tasks are expressible in this model, as shown in the paper. Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system.[5]

3.2 Programming Model

The computation takes a set of input key/value pairs, and produces a set of output key/value pairs. The user of the MapReduce library exhibits the computation as two functions: Map and Reduce. Map, written by the user, takes an input pair in the form of InputFormat (explained above) and produces a set of intermediate key/value pairs. The MapReduce library groups all intermediate values together associated with the same intermediate keyI and passes them to the Reduce function. The Reduce function, also written by the user, accepts an intermediate keyI and a set of values for that key. It merges together these values to form a possibly smaller set of values. Typically just zero or one output value is produced per each Reduce function. The intermediate values are supplied to the user's reduce function via an iterator thus allowing the server to handle lists of values that are too large to fit in memory. [6]

3.3 Class MapReduce

This class supports MapReduce jobs that have multiple input paths with a different InputFormat and Mapper for each path.[7]

addInputPath() Method Detail:

public static void addInputPath (Job job, Path path, Class<? Extends InputFormat> inputFormatClass)

Add a Path with a custom InputFormat to the list of inputs for the map-reduce job.

Parameters:

job - The Job

path - Path to be added to the list of inputs for the job

inputFormatClass – InputFormat class to use for this path

4. DATA ANALYSIS

A sample from New York Stock Exchange Data was analyzed using Java MapReduce framework in Hadoop. [8] MapReduce improves the performance of processing the data by providing automatic data management, fault detection and recovery.

It can help in processing large chunks of data in an economical way. The data in our study is read as input from two files - dividends.csv and price.csv, containing different structures. The amount of data can go upto millions of records. Two mapper classes are employed for both files and one reducer class. After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server. The evaluated output is very accurate. The output achieved consists of lesser records. It can be displayed on three platforms:

i) On Console

ii) On HDFS Files

iii) On Excel sheet.

On Console

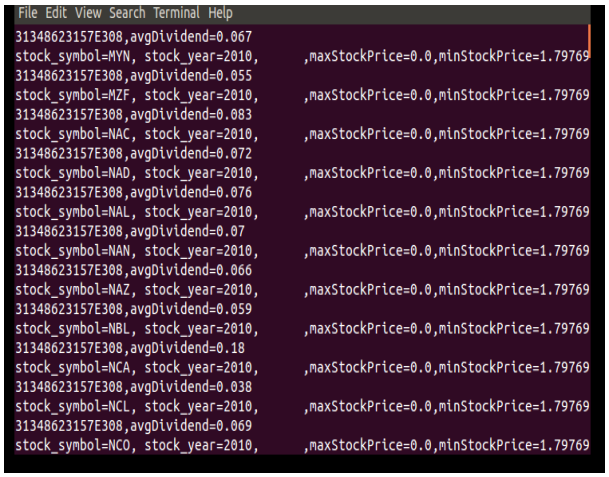

Fig 2: Output on Console screen

On HDFS File

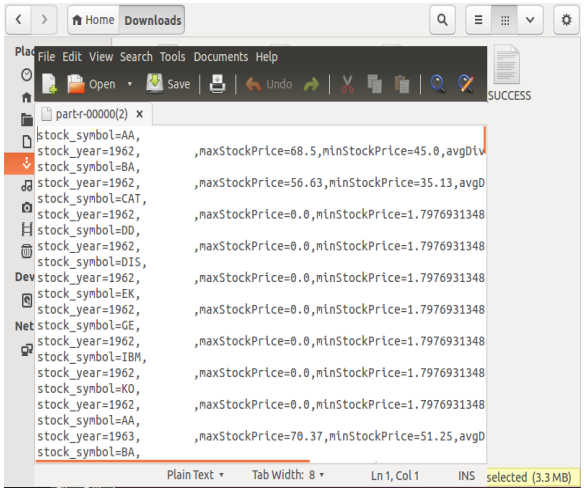

Fig 3: Output on HDFS File

On Excel Sheet

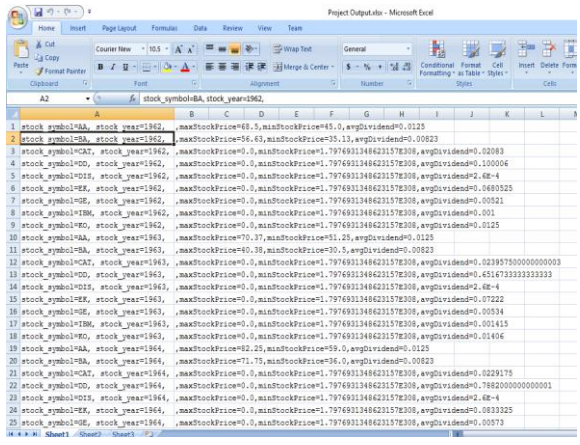

Fig 4: Output on Excel Sheet

5. EXPERIMENTAL RESULTS

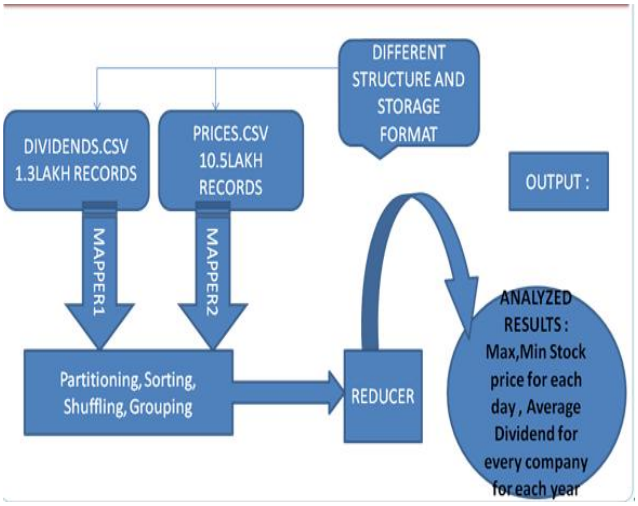

Fig 5: Overview of complete process

The heterogeneous data was successfully processed (see figure 5) and it gave the required computations in very less amount of time (as compared to traditional data processing techniques, see figure 6). The map-reduce process was executed by the Hadoop API (see command to give multiple input path and execution of process in figure 7) and the data files were read, parsed and then processed to give the output file in the Hadoop file system. Two different files having big data were processed together to give one output file. (figure 8) All the files can be browsed in the default URL: http://localhost:50070/.

After the execution of Java map-reduce program two files in the output directory (directory which was specified in the command on console) are received. First file is _SUCCESS file, which is an indicator/flag of successful execution of the program. Second file is the part-r-00000 file which contains the actual output data. If the number of reducers deployed by the processor is more than 1 then separate files are given as output that are named accordingly (eg. Part-r-00001 and so on). Hadoop provides a reliable output in mapping and analyzing of the data. This Data can be employed for further representation.

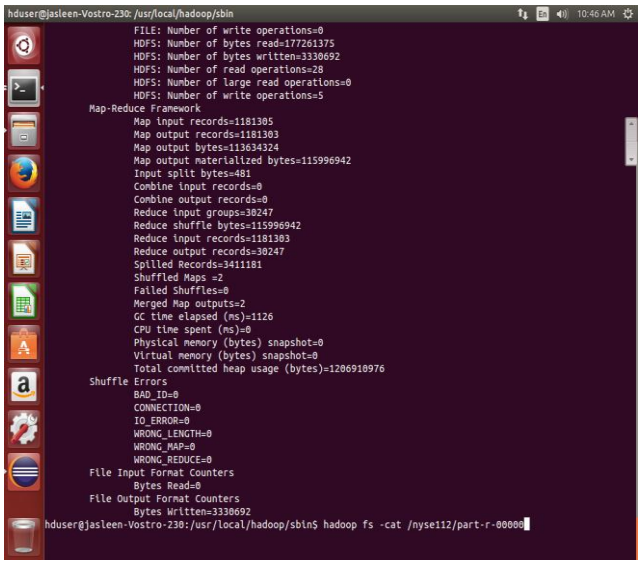

Fig 6: Time taken for processing the data sample.

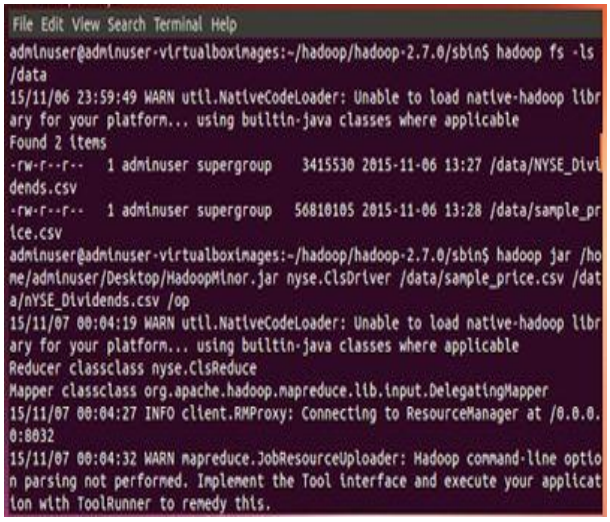

Fig 7: Execution command (multiple inputs were given)

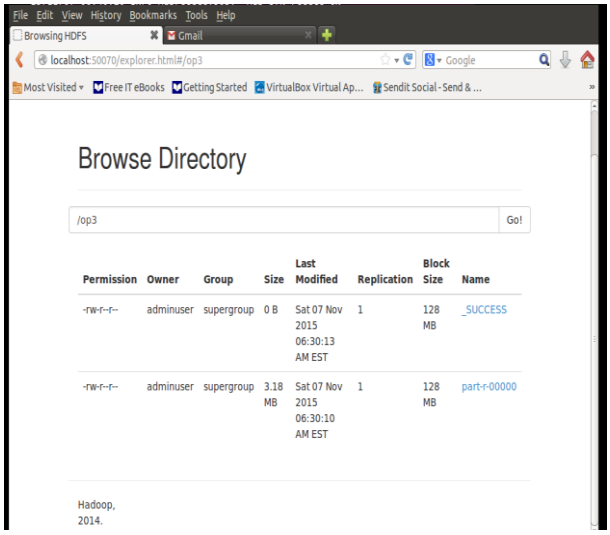

Fig 8: Output directory on web-browser

6. CONCLUSION

Big Data is being captured and used by thousands of organizations and individuals for discovery, research, invention and innovation. It's also being used to make money in new ways previously unimagined or unattainable. The proposed system achieves different parameters of big data.. Big data framework considers complex relationships between samples, models and data sources along with their evolving changes with time and other possible factors. To support big data mining high performance computing platforms are required. With Big data technologies, most relevant and most accurate social sensing feedback can be provided to better understand our society at realtime.[9] There has been a noticeable growth in the usage of the Hadoop from the last decade. Almost all the prominent companies like Yahoo, Google, Amazon, Facebook, IBM etc prefer Hadoop much more over other technologies because of processing of unstructured data being easier in Hadoop than others. There are many companies that are giving up themselves due to the inadaptability of the big data techniques. As of now more than 50% of the fortune 50 companies are using Hadoop.[10]

It can be confidently concluded that processing of voluminous data is less time consuming with Hadoop technology as compared to the time taken by traditional data processing techniques. Heterogeneous Data can also be taken for doing Hadoop data processing. Successful implementation of hadoop framework has been done on Linux operating system and the sample data has been processed according to our needs, using Java map-reduce functionality of Hadoop.

In future, this project can be taken ahead by incorporating different frameworks related to Hadoop like Pig, Hive and Hbase. Sqoop framework can also be used to get the data from HDFS into our Relational database system in NTFS. Also, different type of computations can also be done on input data.

7. REFERENCES

[1] Thomas H. Davenport, 2014 big data @ work Harvard business review press.

[2] T. Kraska, "Finding the Needle in the Big Data Systems Haystack," IEEE Internet Computing, vol. 17, no. 1, pp. 84-86, 2013.

[3] Lekha R.Nair, 2014 Research in Big Data and Analytics: An Overview IJCA Volume 108

[4] Siddharth Mehta 2015 Big Data analytics made easy with SQL and MapReduce

[5] Online Searcher: Information Discovery, Technology, Strategies Volume 38, Number 2 - March/April 2014

[6] J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In Communications of the ACM, 51 (1): 107-113, 2008.

[7] https://hadoop.apache.org/docs/stable/api/org/apache/hadoop/mapreduce/lib/input/MultipleInputs.html

[8] P. Amuthabala Kavya. T.C 2016 Outlook on various scheduling approaches in Hadoop P. Amuthabala et al. / International Journal on Computer Science and Engineering (IJCSE).

[9] Manisha R. Thakare S.W. Mohod A.N. Thakare Various Data-Mining Techniques for Big Data IJCA Number 8

[10] Kvn Krishna Mohan, K Prem Sai Reddy 2016 Efficient Big Data Processing in Hadoop MapReduce IJARCSSE Volume 6 Issue 3