Volume 6, Issue 3, March 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Predictive Analysis of Rainfall Data to Help the Farmers

S. Shajitha Banu, S. Manjula, S. Swathi Priya, V. Yamuna Devi, M. Thangamani
Department of Computer Science, Saranathan College of Engineering, Panjapur, Tiruchirappalli, Tamil Nadu, India

Abstract*— Generating and collecting large datasets is becoming a necessity in domains that also need to keep and process the data for longer periods. Thus, Hadoop system was developed to enable transformation and analysis of vast amount of structured and unstructured data. Weather plays a vital role in agriculture. Rainfall is the major source of water that agriculturist depend on to cultivate their crops. Analyzing the historical data and predicting the future will help the farmers wisely plan their cropping patterns and increase their profit. Hence in our project we did the analysis by finding the average rainfall and inflow in a dam for past 65 years and correlated them to predict the rainfall in forthcoming years with their corresponding inflow to the dam using MapReduce of Hadoop and R programming.

***Keywords*— Map-Reduce, Hadoop, Analysis, Prediction, HDFS.**

I. INTRODUCTION

Histories of weather data such as rainfall and water flow in dams will be huge in size. Performing computation or analysis on such data is a tedious process if we follow traditional analysis approaches.

To overcome those complexities, we used Java MapReduce code to find the average rainfall and inflow of water in a dam and correlated them and performed analysis over the results and predicted the future rainfall and inflow patterns.

Big data is a catch-phrase, meaning a massive volume of both structured and unstructured data that is so large and difficult to process with traditional database techniques Volume, Variety, Velocity, Veracity, the 4V's of Big Data describes its characteristics.

To process such large volume of data Apache introduced Hadoop. Hadoop is an open source java based software framework for storing data and running applications or clusters of commodity hardware. It provides massive storage for any kind of data with enormous processing power and the ability to handle virtually limitless concurrent tasks or jobs.

Apache Hadoop components include MapReduce, HDFS, Pig, Hive and Sqoop. Hadoop can run in three different ways:

1. Standalone mode - Everything is run as a single java process.
2. Pseudo Distributed Mode – Hadoop is run on a single machine with different Hadoop daemons run as different java process.
3. Fully Distributed or Cluster Mode - one machine is the cluster labelled NameNode and another machine is designated as Job Tracker. Rest of machines act as DataNodes and Task Trackers.

Hadoop uses HDFS for data storage. HDFS contains single Name Node and many Data Nodes. By default, Hadoop uses FIFO scheduling and optionally 5 scheduling priorities to schedule jobs from a work queue.

MapReduce is the heart of Hadoop. It is the programming paradigm of Hadoop that allows for massive scalability across hundreds or thousands of servers in a Hadoop cluster. It condenses large volume of data into useful aggregated results. It performs two separate and distinct tasks. The first is Map job which takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value) pairs. The Reduce job takes the output of the map as input and combines those tuples into a smaller set of tuples and performs necessary operation on the tuples and produce the results in the form of a key/value pair.

R is an integrated suite of software facilities for data manipulation, calculation and graphic display. It provides a variety of statistical and graphical techniques and provides effective data handling and storage facilities. The task involved in the project are summarized as

Building an analytical engine to perform analysis over huge data and perform computation over the data(calculating average) using MapReduce.

Predicting or forecasting future pattern using Rprogramming.

II. MOTIVATION

As data is tremendously increasing in meteorological domain, storing and analyzing them becomes a great challenge for environmentalist to take effective actions for future. Agriculture which is greatly dependent on rainfall and corresponding collection of water in a dam will require analysis of those data for framing their cropping and irrigation strategies for future. Wider adoption of skillful weather forecasts, over time horizons of hours to months ahead, offers the potential for more efficient usage of scarce water resources in agricultural applications. Thus we used Hadoop and its MapReduce to effectively store and perform computation over such data which might be helpful for environmental researches and farmers. Accurate forecasting results in better control of water availability, more refined operation of reservoirs and improved hydropower generation.

III. MAPREDUCE OVERVIEW

Fig. 1 shows the components involved in our project. Hadoop's basic operating system platform is Linux. Thus we virtualized the windows operating system by using VMWare 10 and installed Ubuntu 14.04 operating system as guest operating system and ran Hadoop over Ubuntu platform. Data collected from rainfall and water flow are transferred to HDFS in CSV format. The Mapper and Reducer of MapReduce library reads and analyze the data and stores the output in HDFS. Hadoop splits files into large blocks and distributes them across nodes in a cluster. Additional software packages can be installed on top of Hadoop such as Apache Pig, Apache Hive, Apache HBase, Apache Sqoop, etc.

A small Hadoop cluster includes a single master and multiple worker nodes. The master consists of a Job Tracker, Task Tracker, Name Node and Data Node. A slave or worker node acts as both a Data Node and Task Tracker.

HDFS nodes are managed through a dedicated Name Node server to host the file system index and a secondary Name Node can generate snapshots of the Name Nodes memory structures there by preventing file-system corruption and loss of data. Job Tracker can manage job scheduling across node.

Though, MapReduce java code is common, any programming language can be used with "Hadoop Streaming" to implement the "Map" and "Reduce" parts. After the map phase and before the beginning of the reduce phase is a hand off process, named shuffle and sort. Here results from mapper are sorted by key, partitioned if there are multiple reducers and then written to disk. Since data transfer across network is expensive and to limit the volume of data transfer between Map and Reduce tasks, combiner can be used. It summarize the map output records with same key and output of combiner will be send over network to actual reduce task as input. A combiner is also known as a semi-reducer.

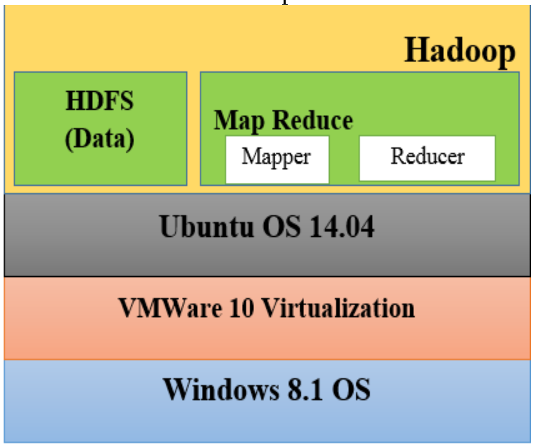


Fig. 1Functional Components

IV. APPROACH

A. System Architecture

The system architecture as in Fig. 2 describes various modules. The MapReduce code to calculate average is submitted to the Job Tracker. The Job Tracker assigns the tasks of Map and Reduce to the Task Tracker. The input file from HDFS is split into blocks of default size 64MB and each block is assigned to a Task Tracker. The Task Tracker's Mapper M1 using Map () function reads the input file and frame them into <key, value> i.e. <year, rainfall> <year, inflow>. The partitioner and combiner group and shuffle the Map outputs and produce it the Reducer R1.The Reducer reads the Map output, sort them and compute the average rainfall and inflow for each key and produce the output key/value pair <year, average> in output file created in HDFS.

The allocation of work to Task Trackers is very simple. Every Task Tracker has a number of available slots(such as "4 slots").Every active map or reduce task takes up one slot. The Job Tracker allocates work to the tracker nearest to the data with an available slot. There is no consideration of the current system load of the allocated machine, and hence its actual availability.

If one Task Tracker is very slow, it can delay the entire MapReduce job-especially towards the end of a job, where everything can end up waiting for the slowest task. With speculative execution enabled, however a single task can be executed on multiple slave nodes.

HDFS stores large files in range of gigabytes to terabytes. It achieves reliability by replicating the data across multiple hosts. Hence does not require RAID storage on hosts. By default data replication value is three. Data is stored in three nodes.

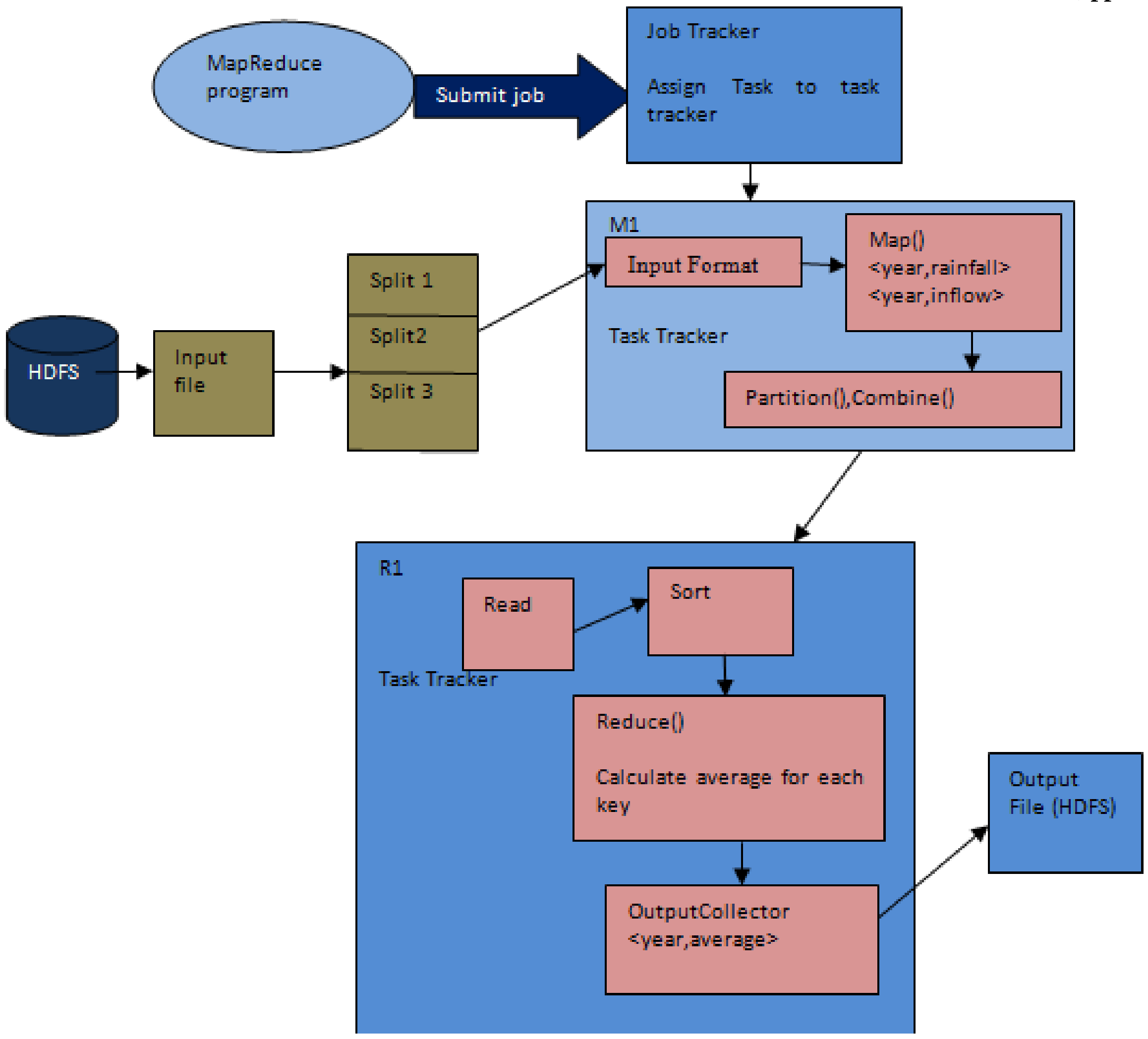


Fig. 2 System Architecture.

B. Analysis using MapReduce

1)Data Flow:

We collected the historical data of rainfall and inflow of water in a dam in the form of structured data. We converted it into a csv (comma separated value) file and stored it initially in local file system. Then, the data from the file system is transferred to the HDFS of Hadoop through the Hadoop commands. .

2)Mapper:

The Mapper reads each record of the input file using the value.toString() and splits each records into fields. The input types fed to the mapper are controlled by the *Input Format* used. The default Input Format is TextInputFormat The data of rainfall contain fields named year, month, rainfall. The data of water inflow contain fields named year, month and inflow of water. The Mapper code is written to read the year as the key and rainfall data and inflow as the value. It then gives the key/value pair <year, rainfall> and <year, inflow> as the output of mapper to the Reducers.

3)Reducer:

Before the Reducer commences its operation, combiner and shuffler groups all the values associated with the particular key and sends the values with the same key K <K, value 1,value 2,..> to the same Reducer.

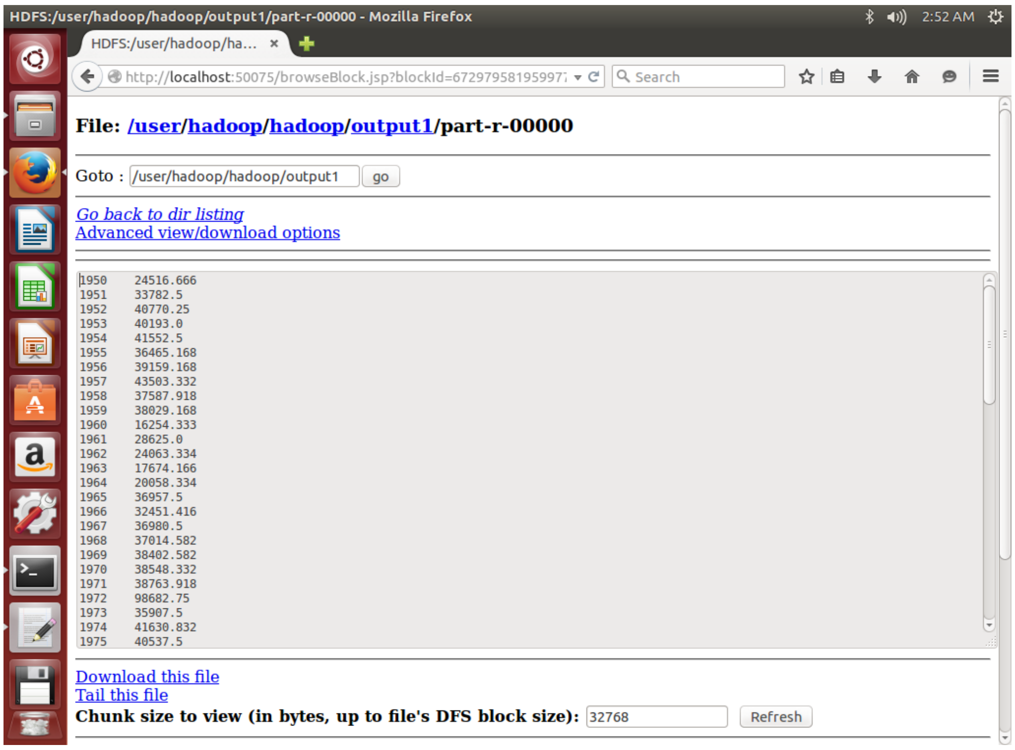
Now the Reducer uses the values to calculate the average. The Reducer takes each value using value.get() function and sum them and divide the sum by its count to find the average. The reducer performs this operation for N number of years available in the input file. Then it outputs the key/value pair of <year, average> for N years.

4)Driver:

The driver initializes the job and instructs the Hadoop platform to execute the code on a set of input files, and controls where the output files are placed. This method sets up a job to execute the mapreduce program across all the files in a given input directory (the InputPath argument). The output from the reducers are written into files in the directory identified by OutputPath. The configuration information to run the job is captured in the *JobConf* object. The mapping and reducing functions are identified by the setMapperClass() and setReducerClass() methods.

5)Output:

The output produced in HDFS is shown in Fig.3. The output of MapReduce is stored in the FS (File System) output directory which is created by Hadoop named part-00000. The history of MapReduce can also be viewed in _logs directory. The output can also be viewed in terminal of Ubuntu.


HDFS:/user/hadoop/hadoop/output1/part-r-00000 - Mozilla Firefox
HDFS:/user/hadoop/ha...
http://localhost:50075/browseBlock.jsp?blockId=67297958195997
Search
2:52 AM

File: /user/hadoop/hadoop/output1/part-r-00000

Goto : /user/hadoop/hadoop/output1 go

Go back to dir listing
Advanced view/download options

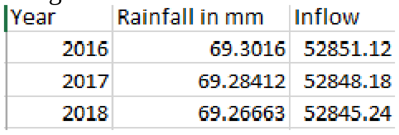
```
1950    24516.666
1951    33782.5
1952    40770.25
1953    40193.0
1954    41552.5
1955    36465.168
1956    39159.168
1957    43503.332
1958    37587.918
1959    38029.168
1960    16254.333
1961    28625.0
1962    24063.334
1963    17674.166
1964    20058.334
1965    36957.5
1966    32451.416
1967    36980.5
1968    37014.582
1969    38402.582
1970    38548.332
1971    38763.918
1972    98682.75
1973    35907.5
1974    41630.832
1975    40537.5
```

Download this file
Tail this file
Chunk size to view (in bytes, up to file's DFS block size): 32768 Refresh

Fig. 3 Output file in HDFS

C. Prediction:

R supports inbuilt function for prediction and graphical representation. The results of MapReduce are downloaded from HDFS and the average rainfall and water inflow in a dam are stored in two separate vectors. The relation between two vectors is defined using the lm() function. Then the predict() function is used to predict the average rainfall and corresponding inflow of water for the next 3 years (2016,2017,2018) based on the relationship between the rainfall and water inflow in dam. The results of prediction are visualized graphically to make decisions based on predicted values which is shown in Fig. 4 & Fig. 5.



Year	Rainfall in mm	Inflow
2016	69.3016	52851.12
2017	69.28412	52848.18
2018	69.26663	52845.24

Fig. 4 Predicted Values

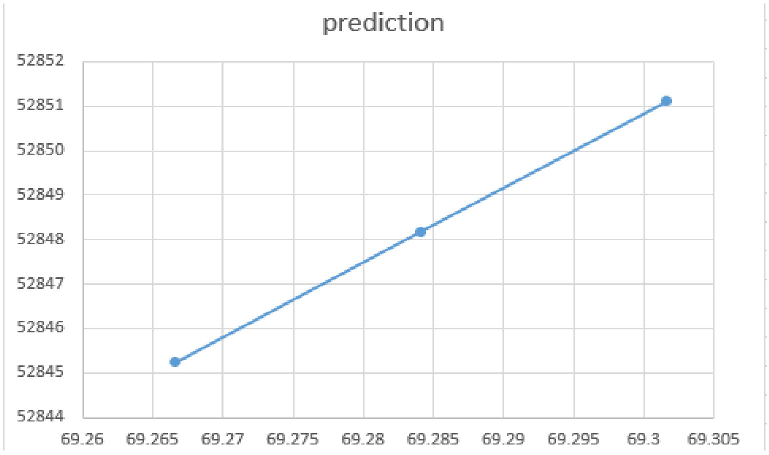


Fig. 5 Graphical Representation of Prediction

V. CONCLUSION

An effective and cheap solution for processing massive volume of meteorological information is highly demanded. In this paper we initiated a basic analysis system to analyse rainfall and water inflow patterns in a dam based on Map Reduce of Hadoop. Further analysis can be done by correlating dam's inflow and outflow of water to warn the people near the banks of the rivers about the floods and to help the farmers effectively utilise the water available from the dams for their irrigation.

REFERENCES

[1] Rainfall Data. Available: http://indiawaterportal.org/.

[2] Hadoop Website. Available: http://hadoop.apache.org/.

[3] *'MapReduce: Simplified Data Analysis of Big Data'*, Seema Maitreya, C.K. Jha, Procedia Computer Science 57 (2015) 563 – 571.

[4] J. Dean and S. Ghemawat, *'MapReduce: Simplified Data Processing on Large Cluster', USENIX OSDI, 2004.*

[5] P. A. Riyaz and Surekha Mariam Varghese: *'Levaraging Map Reduce with Hadoop for Weather Data Analytics',* IOSR Journal of Computer Engineering (IOSR-JCE) e-ISSN: 2278-0661, p-ISSN: 2278-8727, Volume 17, Issue 3, Ver. II (May – Jun. 2015), PP 06-12.

[6] Prashant Shrivastava S. Pandiaraj and Dr J. Jagadeesan, *'Big Data Analytics In Forecasting Lake Levels'*, International Journal of Application or Innovation in Engineering & Management (IJAIEM) ,Volume 3, Issue 3, March 2014.

[7] Victor Sheng, Wei Fang, Wubin Pan, *'Meteorological Data analysis using MapReduce'*, The Scientific Journal February 2014