Storage and Retrieval of Data for Smart City using Hadoop

Ravi Gehlot
Department of Computer Science
Poornima Institute of Engineering and Technology
Jaipur, India

***Abstract*— Smart cities are equipped with a huge range of technologies that generate enormous amount of data. Ubiquitous technologies that are embedded with Radio Frequency Identification (RFID) and Wireless Sensor Network (WSN) that frequently generates data. These devices are implanted everywhere in a smart city. The data generated by the sensors will grow so large, that it cannot be handled by the conventional File Storage Systems. Processing and analyzing of this amount of data needs special tasks to be done. For this we would require a special system that would store and process it with a feasible cost. Hadoop uses Hadoop Distributed File System (HDFS) to store data in distributed servers and MapReduce Algorithm to analyze and process the data. HDFS stores the stream of Application data in Data Nodes, which are mapped by NameNodes. This results in high availability and fault tolerant system. MapReduce algorithm uses key and value pair which makes the analyzing task easier.**

***Keywords*—Hadoop; Hadoop Distributed File System (HDFS); MapReduce; NameNode; DataNode; Big Data**

I. INTRODUCTION

We are generating quintillion bytes of data every day and it has become a tedious task to handle this volume of data. Daily used devices like mobile phones, laptops, computers etc. make use of internet to connect to each other around the globe and in doing so these devices give birth to enormous amount of data. This data is collected at a storage unit now days we call it a Data Center.

The data is growing at an uncontrollable rate and is giving birth to Big Data. We cannot handle Big Data using Relational Database Management System (RDBMS), which uses a tabular representation. Each table in a RDBMS contains rows and columns. RDBMS stores structured data, but the data we receive is not in a structured form. Data we receive, after analyzing the customer's activity, feedback from the end user and other data, is sometimes incomplete and most importantly unstructured. Collecting unstructured data and then storing it using RDBMS is not a feasible approach. So we need to move towards an approach which is more feasible. This paper talks about a solution that is more feasible and more scalable.

Hadoop [1][2] is a fault tolerant and a feasible solution to the problem of Big Data. Hadoop is a combination of multiple things that makes it possible to process structured and unstructured data. Most important of them is HDFS [6]. It is the Distributed File System (DFS) that Hadoop uses to store data and Meta data. It is easier to store and manage data in HDFS then it is in RDBMS. The most important thing of HDFS is it can be implemented on a low cost hardware. HDFS contains two main parts NameNode [6][3] and DataNode [6][3]. DataNodes stores application data on which processing is done while NameNode stores metadata and also NameNodes controls the DataNodes under it. The files that are stored on HDFS are divided into number of blocks (size = 64 MB). NameNode maps each DataNode for Client. And when the required DataNode is found,client contacts it directly.

Processing of data is done using a simple MapReduce Algorithm [6]. This algorithm was launched by Google. Java was used to write Hadoop's framework. Whereas any other language can be used to process data. Map-Reduce Algorithm uses a key and value pair. Before Hadoop came into existence, computations were done by providing data to programs that perform some operations on data and then output was provided. But now, instead of feeding data to the Program, in Hadoop Program is fed to the data. This makes the processing task faster.

Smart cities are loaded with technologies that help people live in a smarter way. There are sensors which controls the traffic, metros and medical anomalies happening in a smart city. You will get reported of a traffic jam on any highway in the city and will also be provided an alternate route. If we talk about devices and sensors residing in the homes of smart cities, they sense the environment and collects data which is uploaded into the servers.

II. RELATED WORK

Work has been carried out to make Hadoop Highly Available [9]. There is a project (*HDFS*-976) [8] of Apache Hadoop that worked on making Hadoop Highly Available. Since all the data that flows in, is controlled by single NameNode (a master node in a single HDFS cluster). This makes it a bottleneck of efficiency and also leads to Single Point Of Failure (SPOF) [5][9] problem. Hadoop provides a secondary NameNode [10] as a solution for this problem which is launched when primary NameNode fails. It reads all the Meta Data from the disks and starts handling the client's request. Similar to this a Backup NameNode was also introduced to make the recovery faster at times of failure. The Backup NameNode [7][9] keeps the updated Meta Data both in disks and ram. But a study by Facebook [11] shows that both the solution consume a lot of time which is not tolerable. So Facebook introduced Avatar NameNode [12][7]. This Avatar Node is like the Secondary NameNode or Backup NameNode that kicks in when the Primary NameNode Fails but it is more faster than both of these NameNodes.

SPOF problem was being deprived off using Quorum Journal Manager (QJM) [5] that comprises of JournalNodes. The Active NameNode transfer its Meta Data to QJM which are logged on

JournalNodes. Giraffa File System [5] which uses HBase, which resolves some more problems like namespace limitations and Load Balancing problem. Just like HDFS, HBase has also one leader i.e. HMaster [5], which continuously monitors the servers. HMaster continuously sends heartbeat singles to the servers. Since HMaster is the single master so its failure cannot be accepted. If HMaster fails then HBase fails and it requires time to recover.

III. HADOOP DISTRIBUTED FILE SYSTEM

HDFS is a Distributed File System (DFS). It is so called because of the reason that it stores data in a distributed manner. HDFS stores data in the form of chunks of size 64 MB. It follows a master-slave architecture. In this architecture slaves can be in thousands and their controlling master is one. Master node is the NameNode that stores Meta Data and help Client [1] to locate data in the slave nodes. It performs some tasks like opening, renaming, and closing data files and data directories. It have some more jobs other than these tasks, which is to map the incoming I/O requests from client so that client can access the data directly from the DataNode. There is a Secondary (backup) NameNode which takes the position of the Primary NameNode when it fails. The Secondary NameNode also maintains a copy of the Meta Data. This Meta Data copy is a mirror image of the Primary NameNode' Meta Data. Sometimes there is a loss of information when failure occurs as the copy of Meta Data when the Secondary NameNode's Meta Data is not that recent.

Slave nodes are known as DataNodes. These DataNodes maintains the Application data. They have different tasks like replication, creation and removal of Data. Files that are received by NameNode are divided into blocks that are replicated on 3 DataNodes to make sure that system is fault tolerant and reliable. Replication tasks are handled by DataNodes only. Task Trackers controls the activities of the DataNodes. Whenever a task is provided by Job Tracker. Tasks on each DataNode are controlled by Task Trackers.

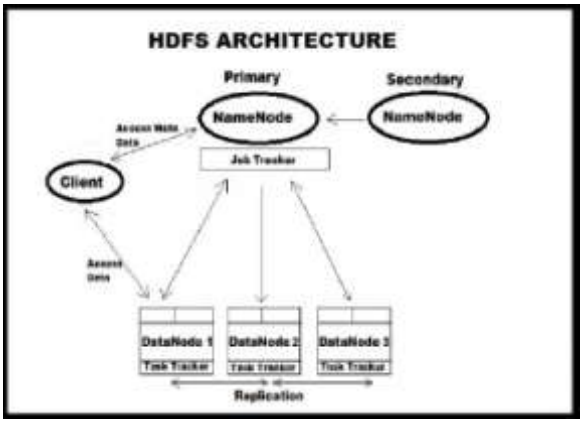

Figure 1. HDFS Architecture

Fig. 1 shows the architecture of HDFS [9][6]. Here client puts his request to NameNode. NameNode search its log files and provide the required DataNode's information to the Client. After getting that information from NameNode, Client goes to the DataNode and retrieve the required information.

IV. MAP REDUCE

Another essential part of Hadoop is MapReduce. The files in HDFS are divided into blocks which are distributed over different DataNodes. MapReduce works in a different way than conventional processing tasks. In this algorithm there are two parts one mapper that maps input data and forms the key value pair and generates intermediate results in a distributed manner. Why distributed, it's because the data reside on different servers. In the second part a reducer program works to rearrange the distributed output and a final output is generated. It works in a different way because in MapReduce computation is done on the server where data is already present. Data is not moved anywhere but the programs that would be processing the data are moved to each location where data is present. Each NameNode is assigned a Job which is tracked by a Job Tracker. This Job is sub divided into many tasks which run locally to a DataNode. These Tasks are tracked by Task Trackers. Task Trackers reports directly to the Job Tracker. After result is generated by these tasks, reducer program combines all the result and provide a final output and is reported to the Job Tracker. In fig.2 we have shown a simple example of algorithm for counting words in java language.

```
map(){
    input(file_name,file_content)
 for each word : file_content
 emit(word,1);
 }
  reduce(){
 input(word,values)
 for each value : values
 sum+=value;
}
```

Figure 2. Algorithm for counting words in a file

As we have already discussed, we used two programs in the above mentioned sample. One mapper that maps the words written in a file and provide a count value to each word. The intermediate result is collected by reducer program and combines all the generated output from the mapper and provides cumulated output.

V. PROPOSED WORK

With the help of the work done previously in the field of Hadoop, this paper provides the solution for storing and managing Big Data generated by the devices used in the Smart City. In this section we would be discussing the efforts that can be made to manage Big Data. First we provide a process that may be followed for acquiring, storing and processing of Data.

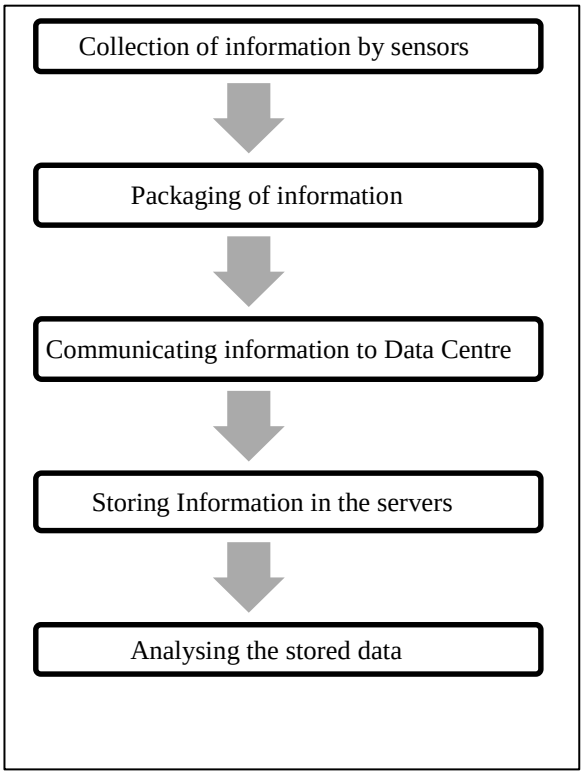

Figure 3. Processing and Storing Data

In fig.3 we have shown a technique to gather the generated information and then storing in a central system.

A. Assembling Data

First step of the process would be collecting the data from the surroundings. This collection process will be

done by sensors. When sensors interact with humans or surrounding they generate data. For e.g. a smart refrigerator submits its owner a daily report that contains the materials that are used on daily basis, like if milk bottle is empty or not, or the food which is present inside should be used further or not. In a smart home most of the appliances that are used are like this smart refrigerator and there are a lot of houses in a Smart City so it is not possible for each house to store this collected data and maintain it for self-use. So we provide a better solution to this. We collect all the data generated from these devices and provide a storage location like Data Centre to store all the data. These centers will make use of Hadoop to deal with the collected data.

B. Transporting Data

When collection is done, data is transported to a Data Center. Whole city would be connected using Light Fidelity (Li-Fi) [13]. Li-Fi is light based communication technology that uses visible light to communicate data. We would be using this service and if not than any other wireless communication services that would be available in the smart city would be used. The data which was collected by the sensors cannot be sent directly into the database. First we require packaging of the data to be done. This packaging would be done just after the data is collected. So we would be requiring a central system in each house that would be controlling each device implanted in the house. This central system would be solely responsible for packaging of data and then sending it for further processing or storing.

C. Processing and Storing of Data

When data is received at Data Centre it is stored in the dedicated servers. For each block or a sector of the city, they will have their dedicated servers. Some of the servers will also store data related to public sectors like data related to traffic, polling, complaints, government buses etc. When a particular user (Here user is the central system of the house controlling all the devices in the city), will put a request for data mining, the Map-Reduce Framework will be responsible for mining tasks.

VI. EVALUATION

In this evaluation we tested a data set of size 1 GB, 2 GB, 3 GB, 4 GB, and 5 GB. We prepared 10 node cluster to test performance of Hadoop. We installed Ubuntu 15.04 on each node. Hadoop version used for this process was 1.2.0 and JDK 8 was used. The data set was distributed over 10 nodes in the chunks of 64 MB on each device.

Figure. 4. Evaluation of Different Data sets.

In this experiment we used 10 nodes and after evaluation we found that as we are increasing the data set size, performance is increasing. But after a threshold the results will decrease because of the less number of nodes. So at that moment we need to increase the nodes in a cluster. Hence the experiment suggest us to use Hadoop for processing the Big Data generated by the Smart Cities.

VII. USE CASES OF HADOOP

The main aim of this section is to support the storing and processing of Data in Hadoop clusters. We need this because it would lead us to certain possibilities that is right now hidden from us. It might help us to raise our living standards. So here we will be discussing different sectors of a smart city where collected information can be used to save money, time and energy also.

A. Metro Stations

People in a city uses a metro more than their personal vehicle. As it has many advantages like it travels fast, it doesn't get stuck in traffic, it has hard deadline and also you maintain a distance from outside heat. But some time there is rush in the metro too. So if we collect the data regarding people travelling in metro and analyze it, then we can find when there is rush. This will help to control metro timings and coaches it carries.

B. Traffic monitoring

Traffic monitoring systems monitor vehicles moving on the road. They recognize easily if any vehicle is breaking traffic rules. These systems also takes data for possible traffic jams and while doing so it also guides other vehicles to change their route. If a road accident occurs then traffic monitoring system can identify it and can call medical help at that time. If these accidents or traffic jam continuously occurs on the same road, then they can be reported as accident prone areas. This would be only possible if the data is collected and managed.

C. Medical Field

There are varieties of disease in this world. People keeps visiting hospitals for their respective health issues. Sometimes it is a matter of life and death. What if a person needs blood or an organ transplant or something like that? In this case maintaining a list of donners could help, but doing this is a tedious task. Now two things can help in this case. Either we can create a database repository for self-use of a particular hospital or we can create a central repository which can include call the hospitals which can save help patient properly.

D. Resource Utilization

Resource utilization needs to be checked. Now we cannot allow their wastage. So to conserve them we have smart devices that controls their use for e.g. if we talk about smart meters that save electricity by checking the flow of current. We have sensors that controls the lights in a room. When any person is not in the room or house, sensors turns them off hence saving energy. These devices also generates data and they also use old data to make their decision. For e.g. at some particular event, consumption of electricity exceptionally increases than its daily use. Now at that moment of time smart electricity meters should be providing enough current to supply and which is again possible only by storing and managing data.

VIII. CONCLUSION

In this paper we initiated from how much data we are generating and moved towards how to manage Big Data. Later we saw that it is easy to manage Big Data with HDFS and to process it using Map Reduce Framework. We discussed about HDFS, how it works and how it makes Hadoop so fast. We learned about Map Reduce Framework, its working and the flow of computation. Working with MapReduce is easy and most importantly Hadoop is not Language depended and it's free. In the end of this paper by proposing a system and provided some use cases of Hadoop in smart city.

References

[1] Apache Hadoop http://haddop.apache.org/

[2] T. White, Hadoop: The Definitive Guide. O'Reilly Media, Yahoo! Press, June 5, 2009.

[3] Konstantin Shvachko, HairongKuang, Sanjay Radia, Robert Chansler Yahoo!, "The Hadoop Distributed File System," IEEE NASA storage conference, http:// storageconference.org/2010/Papers/MSST/ Shvachko.pdf.

[4] ApacheHBase. http://hbase.apache.org/

[5] Yonghwan KIM, Tadashi ARARAGI, Junya NAKAMURA and Toshimitsu MASUZAWA," A Distributed NameNode Cluster for a Highly-Available Hadoop Distributed File System", IEEE 33rd International Symposium, pp. 333-334, 2014

[6] E.Sivaraman, Dr.R.Manickachezian," High Performance and Fault Tolerant Distributed File System for Big Data Storage and Processing using Hadoop", IEEE Intern Conf., pp. 32-36, 2014

[7] Zhanye Wang, Dongsheng Wang, "NCluster: Using Multiple Active Namenodes to Achieve High Availability for HDFS", IEEE Intern Conf., pp. 2291-2297, 2013

[8] Apache Hadoop Project HDFS976, "Hadoop Avatar Node High Availability", http://hadoopblog.blogspot.com/2010/02/hadoopnamen ode-high-availability.htm, February 6, 2010.

[9] Mohammad Asif Khan, Zulfiqar A. Memon,Sajid Khan,"Highly Available Hadoop NameNode Architecture", IEEE Intern Conf., pp. 167 -172, 2012

[10] T. White. Hadoop: The Definitive Guide. O'Reilly Media 2009.

[11]Facebook has the world's largest Hadoop cluster! http://hadoopblog.blogspot.com/2010/05/facebookhasw orlds-largesthadoop.html

[12]D. Borthakur et al. Apache Hadoop Goes Real-time at Facebook. SIGMOD 11: Proceedings of the 2011 International Conference on Management of Data.

[13] About Li-Fi http://www.lifi-centre.com/about-lifi/what-is-li-fi-technology/