An Efficient Model of Distribution and Storage of Big Data across Cloud-based Nodes using Billboard Manager

Rajesh Bose[1], Sudipta Sahana[2], Debabrata Sarddar[3]

ABSTRACT- There are technology and tools that allow complex processing of voluminous data, today. This paper proposes a similar system which aims at easing the process of data storage and retrieval on cloud networks with special focus on providing optimum access over varying Internet connection speeds. The proposed system relies on constantly calculating the closest cloud network depending on factors such as bandwidth available, latency history, storage space level, and closest geographical location. The authors aim to offer the Billboard Manager model to distribute data across the cloud for storing and retrieval of data on a real-time basis using these factors as described in detail in this paper. Further, the proposed Billboard Manager model offers the opportunity to build up an actively indexed set of keywords extracted from Meta data of assorted files passing through it. This enhances the ability of presenting a comprehensive search pattern, thereby, quickly identifying the files and media stored across those cloud networks registered with the Billboard Manager. The ability to plug compression and decompression technology within this proposed Billboard Manager model further enhances the speed at which humongous volumes of data can be distributed across the cloud nodes registered with the Billboard Manager. The design of this model also accommodates parallel processing of data, not entirely dissimilar to existing and popular tools available in the market today. This, therefore, permits Billboard Manager to pre-fetch data by assuming users' needs beforehand. This feature can be advantageous in regions where Internet speeds are considered below par.

Keywords- Cloud computing, Quality of Service, Billboard Manager, Virtualization, Monitoring, Meta data, parallel processing.

———————————— ◆ ————————————

1 Introduction

With the proliferation of internet and rise in standards of computing hardware at affordable rates, organizations and businesses increasingly find themselves empowered to hold and retain increasingly large quantities of data and information for extended periods of time. The ability to do so allows companies to help its employees identify and analyze data to further business interests, identify emerging markets, increase communication, reduce data redundancy, and maximize capital investments. From the days when businesses dealt with only documents, spread sheets to medium-sized database management systems at the most, employees of today deal with multiple types of files ranging from video, audio, blogs, social media content, and real-time streaming data - cumulatively referred to as big data. Newer cloud-based technologies are required to be designed and developed to help in complex concurrent processing. With the advent of big data, cloud computing has become associated with a new prototype for setting aside computing infrastructure for performing several operations on huge volume of data. By itself, the term "big data" is stated to define as that which "Represents the progress of the human cognitive processes, usually includes data sets with sizes beyond the ability of current technology, method and theory to capture, manage, and process the data within a tolerable elapsed time" [1].Gartner, Inc. defines "big data" as "Big Data are high-volume, high-velocity, and/or high-variety information assets that require new forms of processing to enable enhanced decision making, insight discovery and process optimization"[2]. In its simplest form, "big data" is expressed to mean as extremely large bodies of data sets that are so

Rajesh Bose is currently pursuing Ph.D from Kalyani University. He is an IT professional employed as Senior Project Engineer with Simplex Infrastructures Limited, Data Center, Kolkata. He received his degree in M.Tech. in Mobile Communication and Networking from WBUT in 2007. He received his degree in B.E. in Computer Science and Engineering from BPUT in 2004. He has also several global certifications under his belt. These are CCNA, CCNP-BCRAN, and CCA(Citrix Certified Administrator for Citrix Access Gateway 9 Enterprise Edition),CCA(Citrix Certified Administrator for Citrix Xen App 5 for Windows Server 2008).His research interests include cloud computing, wireless communication and networking.

Sudipta Sahana is an assistant professor of a renowned engineering college of west Bengal. More than 4 years he has worked in this region. He has passed his M.tech degree in Software Engineering and B.Tech Degree in Information Technology from west Bengal university of technology with a great CGPA/DGPA on 2010 and 2012 respectively. He is recently working in Ph.D. on the domain of "cloud computing". He is a member of the Computer Science Teachers Association (CSTA), and also a member of International Association of Computer Science and Information Technology (IACSIT).

Debabrata Sarddar, Assistant Professor in the Department of Computer Science and Engineering, University of Kalyani, Kalyani,Nadia, West Bengal, INDIA. He has done PhD at Jadavpur University. He completed his M. Tech in Computer Science & Engineering from DAVV, Indore in 2006, and his B.E in Computer Science & Engineering from NIT, Durgapur in 2001. He has published more than 75 research papers in different journals and conferences. His research interest includes wireless and mobile system and WSN, Cloud computing.

complex that it becomes increasingly difficult to process with available database management systems. Wikimedia describes "big data" as being "In information technology, big data is a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools". Big Data is a data analysis methodology enabled by a new generation of technologies and architecture which support high-velocity data capture, storage, and analysis [3]. There was a time when data sources were deemed to mean traditional or legacy based database systems managed and maintained by corporates to serve their business requirements. Handheld devices and mobile computing power has now thrown open the gates to newer, but unstructured and complex, data sources which have no standard formatting [4]. This has spurred ever-increasing requirements for space. Although the cost of hardware for data storage has kept more or less steady with increasing storage space offerings, organizations and businesses have become all too aware that in order to gain maximum advantage from the data that is being generated continually, they now need to address the challenge of computing power that is needed. Usually, big data storage and physical organization facilities for processing of such data are designed around clustered network-attached storage (NAS) [7]. A collection of NAS "pods" comprising of several storage units connected to each NAS "pod", is required to be configured for clustered NAS infrastructure. The series so connected allow sharing and searching of data on a huge scale [5].Big Data analytic techniques have empowered small to medium-sized organizations to turn to data storage on cloud. Businesses can turn their focus on their core business interests leaving management of on-demand access to computing sources and facilities to cloud service providers [6]. However, small to medium scale organizations need to be constantly alert to their big data processing and storage requirements in order to operate on razor's edge in today's modern competitive environment where costs threaten to engulf profits should there be a lack of active vigilance on the part of the managers. A significant portion of our proposed model of Billboard Manager revolves around applying Map Reduce techniques on extremely large data sets and then distributing the data to cloud nodes that are registered with our proposed Billboard Manager in a manner such that data storage and retrieval are conducted seamlessly and as quickly as possible. The organization of this paper is as follows: In section 2, we will discuss about related works. In section 3, we will discuss about proposed work, in section 4, we will discuss about our proposed algorithm. Experiment verification and result analysis is discuss in section 5, in section 5.1, here we will discuss about Extraction of Metadata and Text from Multimedia content, in section 5.2 we will discuss about Distribution and Identification of cloud storage networks and in section 6 we will discuss about Conclusion & Future work.

2. Related Work

A lot of research has been done in the field of Big Data Processing in Cloud Computing area. In the paper [9] authors introduces several big data processing techniques from system and application aspects. First, from the view of cloud data management and big data processing mechanisms, Authors present the key issues of big data processing, including cloud computing platform, cloud architecture, cloud database and data storage scheme. A good report has been presented in [10].In this paper, authors propose a modified Map Reduce architecture that allows data to be pipelined between operators. This extends the Map Reduce programming model beyond batch processing, and can reduce completion times and improve system utilization for batch jobs as well. Here authors present a modified version of the Hadoop Map Reduce framework that supports online aggregation, which allows users to see "early returns" from a job as it is being computed. In paper [8] authors discuss on two fundamental technologies: distributed data store and complex event processing, and workflow description for distributed data processing.

3. Proposed Work

Our paper is based on a proposed architecture titled "Billboard Manager". It is not aimed at improving upon existing open-source and commercial applications. Nor can it be compared with most popular Big Data frame works. The proposed architecture would work based on a private cloud data center, and offers to transport data to and from registered cloud nodes at an optimum speed, thereby, enhancing user experience wherever possible in regions where Internet speeds are considered slow or inconsistent. The primary objective of this paper is to detect the best fit cloud node - in terms of shortest distance to node, bandwidth availability, and storage space availability - and transmit the data to the designated node as identified by the Billboard Manager. . Figure1 shows the Role of Billboard Manager in our paper. In businesses where extremely large volumes of assorted data - either structured or unstructured - is required to be analyzed using big data analytics, our proposed Billboard Manager model would also enable segregation, aggregation and tabulation of meta data information of assorted media in a manner such that extraction of information is also quicker and economical. A visual representation of Manager the architecture involved of a private cloud data center powered by such
Billboard Manager to support big data ecosystem, is presented in this paper. Billboard Manager consisting of hardware like hard disks, processors, memory modules, solid state disks, secured co-processors, etc. and software [17].Fig2 shows the datacentre architecture for bigdata environment. The Billboard Manager, as would be evident in this paper, would essentially

act as the gatekeeper and manager of all data going in and out through a private cloud computing environment. Data, of assorted types - be it structured or unstructured, from varied sources would be funelled through the Billboard Manager. Sources could be diverse ranging from RDBMS systems, NoSQL stores, information collected from marketing campaigns, advertising and marketing spends, revenue figures generated, and many more. The data passing through Billboard Manager would be analyzed for file type and media signature, identifying of a particular class of content, and for any other user-defined parameters natural to the particular business itself. The very organic nature of the Billboard Manager would allow it to process the data in concurrently running MapReduce processes. The Billboard Manager then looks up a constantly-refreshed set of index cards to identify the clusters, grids, cloud nodes, or even local workstations, powering database manager to which the Billboard Manager can then distribute the compressed data. What is of further interest is the feature of our proposed Billboard Manager model to distribute the aggregated results of the concurrent Map Reduce processes using index cards. Each index card is a constantly-refreshed record set actively maintained and monitored by a separate set of processes or threads running within the Billboard Manager. The refresh rates can be preset or dynamically programmed. However, such refresh rates are not expected to exceed beyond milliseconds for the sake of optimum efficiency in distribution and retrieval of data by Billboard Manager. In essence, the decisive feature of the Billboard Manager is the ability to refresh the set of index cards at extremely high frequency rates. Although this might seem counterproductive given the fact that a significant chunk of resources is constantly tied down, the ability of the Billboard Manager to effectively distribute data, and retrieve such at any point of time quickly, rests on how up to date the set(s) of index cards is / are. A typical index card would hold information on all cloud / cluster / grid nodes (representing assorted database managers at various locations) and local workstations registered with the Billboard Manager. Such

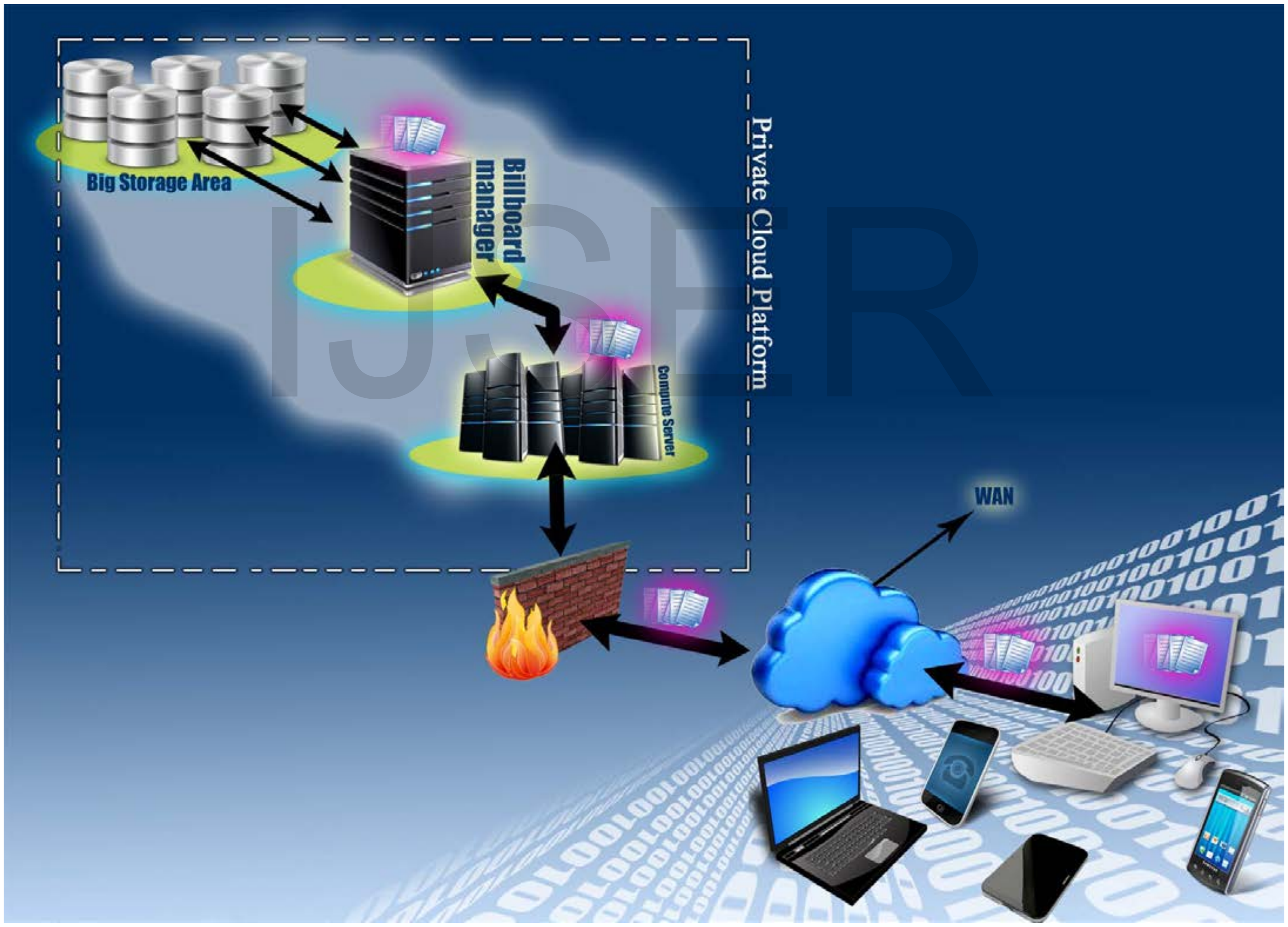

Figure 1. Role of Billboard Manager

information would contain geographical position, storage and channel capacities, latency history, IP addresses, and overall weight index computed periodically at the end of every refresh. For example, a MapReduce parameter could be ascertain the number of keywords appearing in the meta data information of file types being fed into for processing. A meta data or header of several text files can contain, for example, "big data". In several other text files, meta data or headers can contain, say, "algorithm". Assuming that parameters for MapReduce have been based to collate count of unique word,

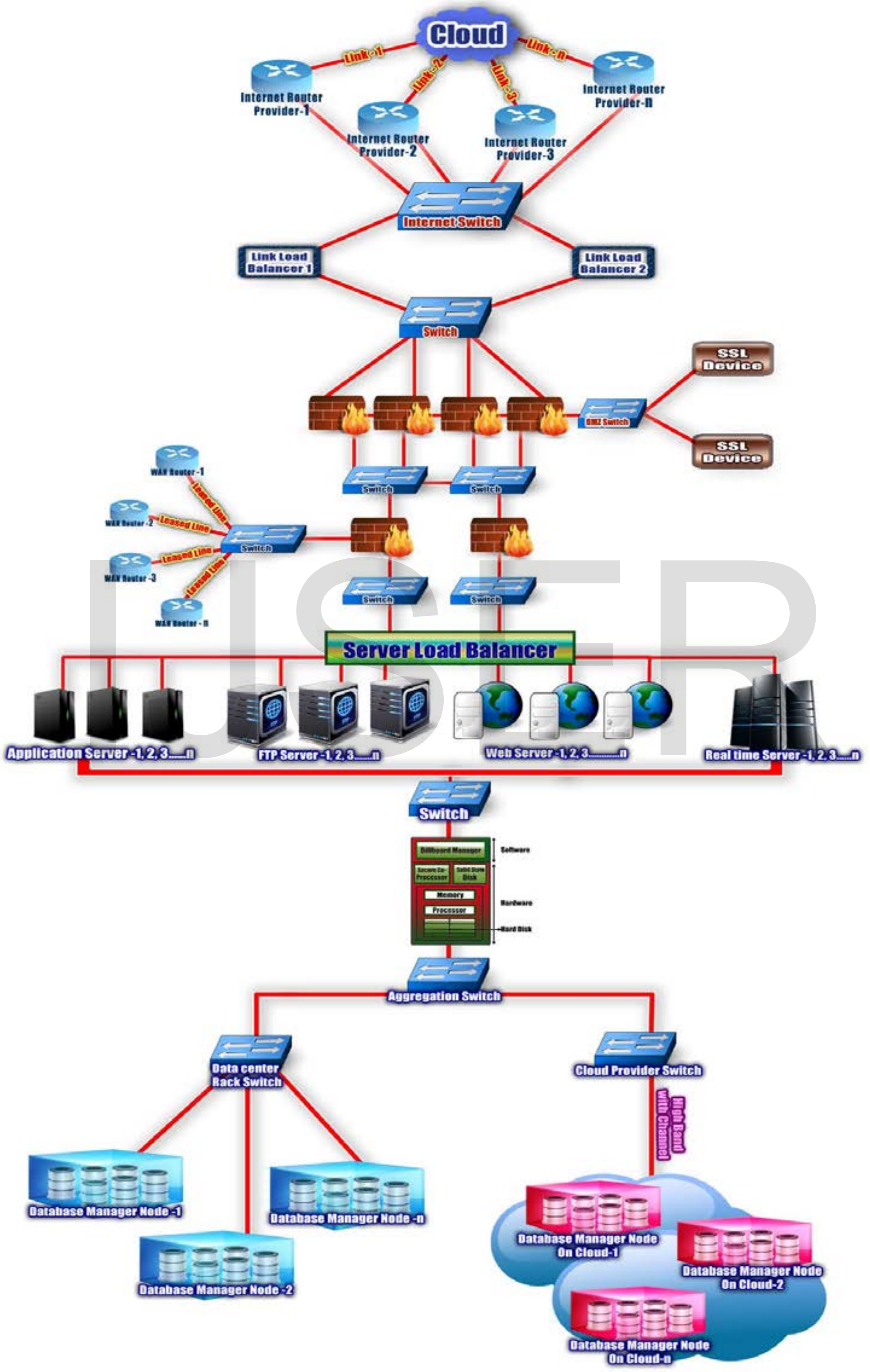

Figure 2. A simple private cloud based data centre architecture for big data environment.

a mapper or MapReduce code would then be able to create a new listing or update an existing one of words and their frequencies. Yet another MapReduce process would maintain a separate list containing file types and the count of each type of file being passed through for processing by the Billboard Manager. With the aid of these two available lists, the Billboard Manager is then able to present a summarized form of search results to the user for a given search string. For example, should a user search for the words "algorithm" and "big", the Billboard Manager should be able to produce a union of results extracted from the first list, merged with that of the second list. Thus, a combined list is presented to a user listing the number of file types available for display containing the words "algorithm" and "big". The proposed Billboard Manager would exist in a public or private cloud. However, to maintain security and confidentiality, a typical business may choose only to host it in a private cloud environment or in a secure server zone. An index card generally would comprise a unique identifying number of the compressed file being deployed for storage by the Billboard Manager. Such index cards will necessarily contain other relevant data such as IP addresses of cloud nodes / workstations, latest ping latency times, geographical location of physical hardware, etc. The purpose of index card is to empower Billboard Manager to quickly locate the file stored in the nearest location to facilitate in easier retrieval when a user makes a selection from the list of results.

4. Algorithm

The Billboard Manager can be fed data which is structured, or otherwise, from different and varied sources. These sources could be as diverse as RDBMS data, NoSQL stores; data collected from marketing campaigns, advertising and marketing spends, figures on revenues collected, numbers of leads generated, weblogs written, user clicks on web pages - to name a few.

The Billboard Manager is primarily tasked to do the following:

a. Scan a particular given directory, file location, or storage medium for files.

b. Detect individual file type.

c. Build an index of file types detected.

d. Scan for meta deta information.

e. Extract words appearing in the meta data information.

f. Correlate that with the index of file types detected.

g. Update and store this index using a designated database manager.

h. Once the meta data and file media types are analyzed and processed, the Billboard Manager looks up a
constantly refreshed index card to identify the clusters, grids, cloud nodes, or even local workstations
running Database Managers to which it could distribute the data.

i. The index card is a record set which is actively maintained and monitored by the Billboard Manager at
preset intervals not exceeding few milliseconds. Since, the Billboard Manager has been designed
keeping in mind that it would need to serve no less than thousands of requests per second, the need to
maintain and keep updating its index card is mandatory. Failure to do so, would slow down the fetch
times in response to requests placed by users to Billboard Manager.

j. The index card is comprised of the following columns:

j.1 Database Manager cluster / grid node identifier.

j.2 Geographical position.

j.3 Capacity - storage.

j.4 Capacity - channel

j.5 Last latency time recorded.

j.6 Latency history weightage computed.

j.7 IP address.

j.8 Overall index.

The higher the overall index, more priority is likely to be assigned by the Billboard Manager when distributing the processed data.

k. The overall index, at a given point in time *T1* is calculated by the Billboard Manager based on the following sequence:

k.1 If storage space at *T1* is equal to or greater than 0, proceed to next step.

k.2 Sort the nodes in descending order of channel capacity.

k.3 Sort the nodes in descending order signal strength.

k.4 Sort the nodes in ascending order of signal latency.

The Billboard Manager compares the columnar values and then builds the initial order of overall index. While building the overall index, the Billboard Manager considers those nodes which are alive. Thus, nodes that fail to respond, or those which are dead, do not qualify for computation of rank overall index at point *T1* when the Billboard Manager updates its index card. For nodes with identical ranks, the one that appears higher, is chosen.

5. Experiment Verification And Result Analysis

The following is an illustration of table of index from a scan conducted on a hypothetical collection of assorted file types which might pass through the proposed Billboard Manager model. It has been assumed that there are only 3 words which appear collectively in the number of files as shown in the table below. However, in a real world scenario there could be thousands of words that can be extracted and tabulated as such.

Sl.	Word	File type: PDF	File type: DWG	File type: XLS	File type: DOC	File type: JPG
1	Billboard	5	0	3	10	20
2	Manager	5	0	3	15	2
3	Big	20	1	20	30	35

Table1: Example of different file types.

Java language (version SE 7) has been used for implementing the fundamental sections of our proposed algorithm. Chiefly, one of the two sections which are of interest is the MapReduce application for aggregation of meta data and media file types. A simplified code that might be used, and example output of this has been reproduced below.

Example of code

```
package File MetaData;
import java.nio.file.spi.FileTypeDetector;
import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;
/**
 *
 * @author SS
 */
public class FileMetaDataDetect {

    String sb1;
    sb1 = Files.probeContentType(p1);
    return sb1;
  }
  public static void main(String... ar) throws IOException {
    Map<String, Integer> m1 = new HashMap<String, Integer>();
    String s1 = "//TestStorage";
    Integer i1=0;
        File location01 = new File(s1);
    for (File f1 : location01.listFiles()) {
      Path p1 = Paths.get(f1.getAbsolutePath());
          if((m1.get(FileMetaDataDetect.Tester01(p1)))==null){
        m1.put(FileMetaDataDetect.Tester01(p1), 1);
      }else{
        i1=m1.get(FileMetaDataDetect.Tester01(p1));
        i1++;
        m1.put(FileMetaDataDetect.Tester01(p1), i1);
      }
    }
    for(Map.Entry s2:m1.entrySet()){
      System.out.println("Count for file type : "+s2.getKey()+" is "+s2.getValue());
    }
  }
}
```

Explanation of code:

The "//TestStorage" is the representation of the directory or storage location from where the files and/or media would be read in by our proposed Billboard Manager model. For simplicity, the code has been built around file directory path design. Subsequent, re-designed iterations of the same code, may employ buffered input streams or other suitable appropriate Java I/O classes as additional or alternate data reading modes.

Resultant Output:

```
Count for file type : image/png is 1
Count for file type : image/jpeg is 3
Count for file type : application/pdf is 1
Count for file type : image/vnd.dwg is 1
Count for file type : image/x-xcf is 1
Count for file type : application/x-php is 6
Count for file type : application/wps-office.doc is 5
Count for file type : application/x-compressed-tar is 1
Count for file type : application/x-trash is 2
Count for file type : application/x-sqlite3 is 1
Count for file type : application/vnd.oasis.opendocument.text is 57
Count for file type : text/html is 13
Count for file type : application/javascript is 1
Count for file type : application/wps-office.docx is 6
Count for file type : application/vnd.oasis.opendocument.spreadsheet is 10
Count for file type: text/plain is 1
```

5.1. Extraction of Metadata and Text from Multimedia Content

However, by far the most important part would be to extract meta data from a wide-ranging types of files. We propose to use the File Information Tool Set [13, 14]. The FITS is a library which acts as a wrapper for assorted file types and enables extraction of metadata from almost all known file formats. While discussion on intricate details on meta data and content extraction is beyond the scope of this paper, future work can encompass extracting rapid extraction of text from videos and images [15]. As organizations and business make rapid strides towards streamlining harvesting of data from their own networks, our proposed Billboard Manager would be able to arm them with text data analyzed from videos and images captured by digital cameras, phones and hand-held computing devices, laptops, surveillance cameras, scanners, etc.

5.2. Distribution and Identification of Cloud Storage Networks

The network index card building algorithm is the other section which is of importance in the context of this paper. Following aggregation of data, our proposed Billboard Manager model would look up its index cards to ascertain which cloud locations would be used to store the data and processed information. Following aggregation of the above data, the information would be processed through the next crucial stage of our proposed Billboard Manager model. This stage prepares the conduit which enables our proposed model to distribute huge volumes of data across its registered cloud nodes. What sets it apart is its ability to narrow down on the most easily accessible cloud node such that the minimum amount of hardware resources is used up while handling extremely large volumes of data storage and retrieval. We discuss next how the proposed Billboard Manager model initiates and maintains processes which keep monitoring ping response times, ping and latency histories of all data centers registered with it across clouds, clusters, WAN, LAN and workstations. From the information gleaned, our proposed Billboard Manager prepares a set of network index cards which would, at any given time, assist the Billboard Manager to direct data storage and retrieval traffic in the quickest manner possible. The Billboard Manager sets up threads to begin processing on index cards which hold real time values of network statistics vital to ensure that there is minimal lag in posting / storing data in a distributed model such as clouds, and retrieving information when such is necessary in the quickest possible times. The threads which would be activated by the Billboard Manager would run full time on a 24x7 basis. These would continue running without a pause until scheduled upgrades are required, or become necessary for the entire design to be replaced by next generation model. At the very initialization of the Billboard Manager, there would be a set number of threads initiated and commenced for every corresponding registered cloud node. It is assumed, that each of those threads would correspond to the following regional internet registries [RIR], viz. [11]:

African Network Information Centre (AfriNIC)
American Registry for Internet Numbers (ARIN)
Asia-Pacific Network Information Centre (APNIC)
Latin American and Caribbean Internet Address Registry (LACNIC)
RIPE Network Coordination Centre (RIPE NCC)

In order to arrive at a standard reference point with which to determine the refresh rate at which each Billboard Manager thread updates its own network index cards or tables, the following assumptions and references are used [12]:

Circumference of the earth: 40075.00 Kms.
Speed of light in vacuum: 299792 Kms/s, approximately.
Reduction in speed of light in Fiber Optic cables: 35%
Effective speed of light in Fiber Optic cables: 194864.80 Kms./s.
Ideal time required to circumnavigate the globe: 40075.00 / 194864.80 = 0.205 seconds.

It is suggested that this time [T1] of 205 milliseconds be doubled and set as the suggested standard limit after which each thread should update ping response and latency times for a given IP address of a nodal reference registered with it. To illustrate, in brief, the workings of this stage of our proposed Billboard Manager model, the following code written in Java version 7 is suggested.

```
package Thread_Ping;
import java.net.Inet4Address;
import java.net.InetAddress;
import java.net.UnknownHostException;
import java.util.HashMap;
import java.util.Map;
import java.util.logging.Level;
import java.util.logging.Logger;
/**
 *
 * @author Rajesh Bose
 */
public class Thread_Ping{
   public static final long CIRCUMNAVIGATE_GLOBE_TIME_MS = 205;
   static void MessagePrinter(String RegionalInternetRegistry, long
timeDiff, InetAddress ipAddr) {
      System.out.println("Time to resolve is " + timeDiff + "ms for " +
RegionalInternetRegistry + ". IP --> " + ipAddr.toString().split("/")[1]);
   }
   static void MessagePrinter(String msg) {
      System.out.println(msg);
   }
   public static class PingTest implements Runnable {
      String RegionalInternetRegistry;
      String hostName;
      InetAddress inA;
      long startTime = System.currentTimeMillis();
      long endTime = System.currentTimeMillis();
      long intervalTime;
      PingTest(String RegionalInternetRegistry, String hostName) {
         this.RegionalInternetRegistry = RegionalInternetRegistry;
         this.hostName = hostName;
      }
      @Override
      public String toString() {
         return "RegionalInternetRegistry=" + RegionalInternetRegistry;
      }
      @Override
      public void run() {
         try {
            startTime = System.currentTimeMillis();
            inA = InetAddress.getByName(hostName);
            endTime = System.currentTimeMillis();
            intervalTime = endTime - startTime;
            MessagePrinter(RegionalInternetRegistry, intervalTime, inA);
         } catch (UnknownHostException ex) {
            MessagePrinter("Unknown host error.");
         }
      }
   }
   public static void main(String ar[]) throws InterruptedException {
      MessagePrinter("**** Beginning tabulating times to resolve hosts /
nodes. ****");
      long startMain = System.currentTimeMillis();
```

```
    long endMain = System.currentTimeMillis();
    long intervalMain = endMain - startMain;
    String hstNames[][] = new String[24][2];
    /* Host name address details obtained from:
     * http://www.pool.ntp.org/en/
     */
        //Input hard-coded in the program. Input can be read off from a
text or binary file as well.
    hstNames[0][0] = "Africa0";
    hstNames[0][1] = "0.africa.pool.ntp.org";
    hstNames[1][0] = "Africa1";
    hstNames[1][1] = "1.africa.pool.ntp.org";
    hstNames[2][0] = "Africa2";
    hstNames[2][1] = "2.africa.pool.ntp.org";
    hstNames[3][0] = "Africa3";
    hstNames[3][1] = "3.africa.pool.ntp.org";
    hstNames[4][0] = "Europe0";
    hstNames[4][1] = "0.europe.pool.ntp.org";
    hstNames[5][0] = "Europe1";
    hstNames[5][1] = "1.europe.pool.ntp.org";
    hstNames[6][0] = "Europe2";
    hstNames[6][1] = "2.europe.pool.ntp.org";
    hstNames[7][0] = "Europe3";
    hstNames[7][1] = "3.europe.pool.ntp.org";
    hstNames[8][0] = "Asia0";
    hstNames[8][1] = "0.asia.pool.ntp.org";
    hstNames[9][0] = "Asia1";
    hstNames[9][1] = "1.asia.pool.ntp.org";
    hstNames[10][0] = "Asia2";
    hstNames[10][1] = "2.asia.pool.ntp.org";
    hstNames[11][0] = "Asia3";
    hstNames[11][1] = "3.asia.pool.ntp.org";
    hstNames[12][0] = "North America0";
    hstNames[12][1] = "0.north-america.pool.ntp.org";
    hstNames[13][0] = "North America1";
    hstNames[13][1] = "1.north-america.pool.ntp.org";
    hstNames[14][0] = "North America2";
    hstNames[14][1] = "2.north-america.pool.ntp.org";
    hstNames[15][0] = "North America3";
    hstNames[15][1] = "3.north-america.pool.ntp.org";
    hstNames[16][0] = "Oceania0";
    hstNames[16][1] = "0.oceania.pool.ntp.org";
    hstNames[17][0] = "Oceania1";
    hstNames[17][1] = "1.oceania.pool.ntp.org";
    hstNames[18][0] = "Oceania2";
    hstNames[18][1] = "2.oceania.pool.ntp.org";
    hstNames[19][0] = "Oceania3";
    hstNames[19][1] = "3.oceania.pool.ntp.org";
    hstNames[20][0] = "South America0";
    hstNames[20][1] = "0.south-america.pool.ntp.org";
    hstNames[21][0] = "South America1";
    hstNames[21][1] = "1.south-america.pool.ntp.org";
    hstNames[22][0] = "South America2";
    hstNames[22][1] = "2.south-america.pool.ntp.org";
    hstNames[23][0] = "South America3";
    hstNames[23][1] = "3.south-america.pool.ntp.org";
    Thread thrdarr[] = new Thread[hstNames.length];
    for(int count=0;count<hstNames.length;count++)
{
        thrdarr[count] = new Thread(new
PingTest(hstNames[count][0],hstNames[count][1]));
    }
    for (int i = 0; i < thrdarr.length; i++) {
        thrdarr[i].start();
    }
    while (intervalMain < CIRCUMNAVIGATE_GLOBE_TIME_MS * 2) {
        intervalMain = System.currentTimeMillis() - startMain;
    }
    for (int i = 0; i < thrdarr.length; i++) {
        if (thrdarr[i].isAlive()) {
            MessagePrinter("Not yet responded, this node -- > " +
hstNames[i][1] + ".");
        }
    }
    System.out.println("\n**** Total process run time ends after : " +
intervalMain + "ms. ****\n");
    System.exit(0);
  }
}
```

Explanation of code:

Considering that active cloud nodes cannot be produced for the purpose of demonstrating this stage of our proposed Billboard Manager, a selection of twenty four host names have been selected [16]. This selection of 24 time servers would help illustrate how our proposed Billboard Manager would go on to build and maintain network index cards. At first, an array of 24 elements is created containing the time server name and host address. This array is then passed through a loop which starts threads corresponding to each of the 24 host addresses. Each thread marks the time it takes to resolve the respective host and its associated IP address. The whole process is timed to end at 410 milliseconds, i.e., 2 x 205 milliseconds. the threads which have not yet completed resolving the IP addresses, are printed out. Subsequently, the hosts which fail to respond within the suggested 410 milliseconds mark would be moved from the primary network index card to the secondary index card.

Sample output #1:

```
**** Beginning tabulating times to resolve hosts / nodes. ****
Time to resolve is 148ms for South America3. IP --> 200.192.232.8
Time to resolve is 200ms for Europe3. IP --> 91.227.249.51
Time to resolve is 201ms for Oceania3. IP --> 202.127.210.36
Time to resolve is 214ms for Europe0. IP --> 83.137.98.96
Time to resolve is 215ms for Africa1. IP --> 196.25.1.1
Time to resolve is 211ms for Oceania1. IP --> 202.22.158.31
Time to resolve is 229ms for North America3. IP --> 70.35.113.44
Time to resolve is 240ms for Asia2. IP --> 120.88.46.10
Time to resolve is 241ms for North America0. IP --> 129.250.35.250
Time to resolve is 260ms for South America0. IP --> 200.192.112.8
Time to resolve is 261ms for Asia1. IP --> 78.111.50.1
Time to resolve is 262ms for Asia3. IP --> 185.23.153.237
Time to resolve is 285ms for Africa2. IP --> 196.43.1.9
Time to resolve is 285ms for Africa0. IP --> 41.231.53.4
Time to resolve is 285ms for Europe1. IP --> 212.83.184.186
Time to resolve is 282ms for Oceania0. IP --> 58.96.157.123
Not yet responded, this node -- > 3.africa.pool.ntp.org.
Not yet responded, this node -- > 2.europe.pool.ntp.org.
Not yet responded, this node -- > 0.asia.pool.ntp.org.
Not yet responded, this node -- > 1.north-america.pool.ntp.org.
Not yet responded, this node**** Beginning tabulating times to resolve
hosts / nodes. ****
Time to resolve is 148ms for South America3. IP --> 200.192.232.8
Time to resolve is 200ms for Europe3. IP --> 91.227.249.51
Time to resolve is 201ms for Oceania3. IP --> 202.127.210.36
Time to resolve is 214ms for Europe0. IP --> 83.137.98.96
Time to resolve is 215ms for Africa1. IP --> 196.25.1.1
Time to resolve is 211ms for Oceania1. IP --> 202.22.158.31
Time to resolve is 229ms for North America3. IP --> 70.35.113.44
Time to resolve is 240ms for Asia2. IP --> 120.88.46.10
Time to resolve is 241ms for North America0. IP --> 129.250.35.250
Time to resolve is 260ms for South America0. IP --> 200.192.112.8
Time to resolve is 261ms for Asia1. IP --> 78.111.50.1
Time to resolve is 262ms for Asia3. IP --> 185.23.153.237
Time to resolve is 285ms for Africa2. IP --> 196.43.1.9
Time to resolve is 285ms for Africa0. IP --> 41.231.53.4
Time to resolve is 285ms for Europe1. IP --> 212.83.184.186
Time to resolve is 282ms for Oceania0. IP --> 58.96.157.123
Not yet responded, this node -- > 3.africa.pool.ntp.org.
Not yet responded, this node -- > 2.europe.pool.ntp.org.
Not yet responded, this node -- > 0.asia.pool.ntp.org.
Not yet responded, this node -- > 1.north-america.pool.ntp.org.
Not yet responded, this node -- > 2.north-america.pool.ntp.org.
Not yet responded, this node -- > 2.oceania.pool.ntp.org.
Not yet responded, this node -- > 1.south-america.pool.ntp.org.
Not yet responded, this node -- > 2.south-america.pool.ntp.org.
**** Total process run time ends after: 410ms. **** -- > 2.north-
america.pool.ntp.org.
Not yet responded, this node -- > 2.oceania.pool.ntp.org.
Not yet responded, this node -- > 1.south-america.pool.ntp.org.
Not yet responded, this node -- > 2.south-america.pool.ntp.org.
**** Total process run time ends after: 410ms. ****
Sample output #2:
**** Beginning tabulating times to resolve hosts / nodes. ****
Time to resolve is 137ms for Africa0. IP --> 41.188.33.6
Time to resolve is 143ms for Asia0. IP --> 202.65.114.202
Time to resolve is 142ms for Asia2. IP --> 120.119.28.1
Time to resolve is 155ms for Africa3. IP --> 196.25.1.9
Time to resolve is 163ms for Europe3. IP --> 176.31.253.84
Time to resolve is 165ms for Europe1. IP --> 85.234.197.1
Time to resolve is 170ms for North America1. IP --> 54.235.96.196
Time to resolve is 164ms for Oceania1. IP --> 27.116.36.36
Time to resolve is 184ms for Europe2. IP --> 5.34.248.224
Time to resolve is 174ms for Oceania2. IP --> 175.45.85.97
Time to resolve is 191ms for Asia1. IP --> 211.233.40.78
Time to resolve is 195ms for Africa2. IP --> 168.167.71.131
Time to resolve is 187ms for Oceania0. IP --> 192.189.54.17
Time to resolve is 212ms for Asia3. IP --> 59.106.180.168
Time to resolve is 207ms for North America2. IP --> 69.50.219.51
Time to resolve is 208ms for South America0. IP --> 146.164.53.65
Time to resolve is 208ms for South America1. IP --> 200.1.19.16
Time to resolve is 215ms for North America0. IP --> 173.230.144.109
Time to resolve is 234ms for Europe0. IP --> 93.184.71.155
Time to resolve is 221ms for South America3. IP --> 200.37.61.2
Time to resolve is 221ms for South America2. IP --> 200.1.19.4
Time to resolve is 234ms for Oceania3. IP --> 116.68.13.155
Time to resolve is 315ms for North America3. IP --> 198.100.156.225
Time to resolve is 337ms for Africa1. IP --> 41.216.204.3
**** Total process run time ends after: 410ms. ****
```

It is suggested that the same process of our proposed Billboard Manager would actively monitor the secondary index card for responsiveness. Should any host be found responding within 410 milliseconds, that would be brought over onto the primary index card.

6. Conclusion & Future Work

Although, our proposed Billboard Manager model resembles existing programming models designed around parallel distribution design in certain aspects, it is by no means intended to be a replacement for such. Our work focuses on selecting the best-fit cloud node based on an index card model that constantly monitors factors such as shortest distance, latency, bandwidth, and available storage. Simultaneously, our Billboard Manager also builds up an index of words extracted from data and media which are to be stored at the appropriately selected cloud node. While such cloud nodes could be a combination of private and public clouds, this proposed Billboard Manager has been conceived as one which would aid not only in bringing forth the appropriate information from a business's cloud storage network, but analyze and store huge volumes of data in a way that would help in making data and information access and retrieval considerably cheaper. While Internet bandwidth is high in most advanced nations, there exist significantly large sections of the globe which are served by inferior quality of connectivity. In such situations, this proposed Billboard Manager model can help businesses in those geographical locations to survive and be cost-effective by constantly building a profile of the data they would be required to deal with. Further, the network index card management of the proposed Billboard Manager would be able to assist in identifying the cloud nodes which are most readily accessible. This would help eliminate cutting down costs in terms of time that would been otherwise spent in sequentially or randomly going through the list of registered cloud nodes from where data can be extracted and then presented in response to the user's search requests. Future work on our proposed Billboard Manager model could revolve around refining the process of extracting content from the data passing through the gateway of the Billboard Manager. Such an extraction of data would be made as quickly as possible without compromising on conciseness of the information extracted. Compression algorithms would be employed to not only compress the data, but also the aggregation of key information extracted from such. The compressed data would be subsequently distributed among the cloud nodes registered with our proposed Billboard Manager. Another important facet of this proposed model that is significant is the use of network index cards. This suggested model of choosing the cloud node which would allow transmission of information within the shortest duration possible, may be considered as the key which is pivotal in the push towards a more greener and economical operation. At a later time, the process of building and maintaining such network index cards could be improved upon to allow transmission of compressed information to cloud nodes selected at a faster rate. Such would depend on the ability to break chunks of the compressed data and distributing these across cloud nodes so that a redundancy level can be maintained without compromising on performance. As a design which could possibly bring down costs further for businesses striving to stay competitive by aligning their resources and investments in information technology, this proposed Billboard Manager Model, and its future iterations, can help organizations and individuals achieve just that goal.

REFERENCES

[1] “Big data: science in the petabyte era,” Nature 455 (7209): 1, 2008.
[2] Douglas and Laney, “The importance of ‘big data’: A definition,” 2008.
[3] Villars, R. L., Olofson, C. W., & Eastwood, M. (2011, June). Big data: What it is and why you should care. IDC White Paper. Framingham, MA: IDC.
[4] Coronel, C., Morris, S., & Rob, P. (2013). Database Systems: Design, Implementation, and Management, (10 th Ed.). Boston: Cengage Learning.
[5] Villars, R. L., Olofson, C. W., & Eastwood, M. (2011, June). Big data: What it is and why you should care. IDC White Paper. Framingham, MA: IDC.
[6] IOS Press. (2011). Guidelines on security and privacy in public cloud computing. Journal of E-Governance, 34 149-151. DOI: 10.3233/GOV-2011-0271.
[7] Sliwa, C. (2011, June 16). Scale-out NAS, object storage, cloud gateways replacing traditional NAS. Retrieved from http://searchstorage.techtarget.com/feature/Scale-out-NAS-object-storage-cloud-gateways-replacing-file-storage.
[8] Satoshi Tsuchiya ,Vivian Lee, Yoshinori Sakamoto, Yuichi Tsuchimoto, “Big Data Processing in Cloud Environments” FUJITSU Sci. Tech. J., Vol. 48, No. 2, pp. 159–168 (April 2012).
[9] Changqing Ji , Yu Li, Wenming Qiu, Uchechukwu Awada, Keqiu Li, Big Data Processing in Cloud Computing Environments, 2012 International Symposium on Pervasive Systems, Algorithms and Networks.
[10] Tyson Condie, Neil Conway, Peter Alvaro, Joseph M. Hellerstein, MapReduce Online,Yahoo! Research
[11] https://www.isoc.org/briefings/021/
[12] http://www.nature.com/nphoton/journal/v7/n4/full/nphoton.2013.45.html.
[13] http://code.google.com/p/fits/
[14] http://projects.iq.harvard.edu/fits/
[15] Wong, E.K. , Minya Chen , A robust algorithm for text extraction in colour video, Multimedia and Expo, 2000. ICME 2000. 2000 IEEE International Conference ,797 - 800 vol.
[16] http://www.pool.ntp.org
[17] Debabrata Sarddar, Rajesh Bose, Creating a Secured Cloud Based Data Center Using Billboard Manager (BM) and Secure Co-Processor, International Journal of Scientific & Engineering Research, Volume 4, Issue 12, December-2013.