International Journal of Computational Intelligence Research
ISSN 0973-1873 Volume 13, Number 5 (2017), pp. 1201-1227
© Research India Publications
http://www.ripublication.com

Handling Complex Heterogeneous Healthcare Big Data

C. S. Sindhu
Research Scholar, JNIAS, Hyderabad & Faculty, Department of CSE, Global Academy of Technology, Bangalore, India.

Nagaratna P. Hegde
Professor, Dept. of Computer Science & Engineering, Vasavi College of Engineering, Hyderabad, India.

Abstract

With enormous amount of heterogeneous Big Data generated worldwide, the need of managing those data efficiently and in a collaborative way has become the need of the hour in the field of data science and technology. Big Data is massive growth of heterogeneous data which can only be managed by the distributed and parallel computing systems and databases globally. There are various limitations associated with existing relational databases for handling a massive amount of heterogeneous data since most of them are customized for a particular format. For optimizing this challenging scenario Hadoop ecosystem has been brought into operation but the distributed file management systems of Hadoop such as HBase also have their limitation in handling the structured format of the data. This paper tries to mitigate one of the problems associated with Big Data , i.e. handling data heterogeneity. We have tried to provide a common framework to store unstructured (txt file formats), semi structured (XML file formats) and structured (XLS format) data into HBase. A text mining mechanism is proposed for building this framework. In the proposed system we have attempted in using three different algorithmic forms which are 1) Centralized Structured data to Distributed Structured data, 2) Unstructured to Structured format and 3) Semi structured to Structured format respectively. The performance analysis is done on XLS data format, an ALS(Amyotrophic Lateral Sclerosis) dataset is used. A corresponding synthetic dataset is created for evaluating unstructured and semi structured data set.

Keywords: BigData, Distributed File Management, Hadoop, Unstructured Data, Semi-Structured Data.

INTRODUCTION

The present business operation of the healthcare industry has undergone a paradigm shift in technological adoption compared to what it was just five years back. Numerous hospitals and healthcare units are connected in a collaborative manner to process huge amounts of complex data. Now a day's various enterprises and commercial firms have reached a situation where data explosion brings out many challenges with respect to rapid growth of information storage systems. The distributed and parallel computing systems more often deal with huge amounts of Data but the existing traditional techniques are not sufficient to process this volume. Bihl, Young II, & Weckman (2016) have examined the various methods of analyzing Big Data, stated the differences between data and Big Data which has lead to easy understanding of Big Data.

It can be seen that BigData processing requires parallel software's which are running at the same time in thousand of distributed server computers (Assunção, Calheiros, S. Bianchi, Netto, & Buyya, 2015). A large set of heterogeneous data includes various type of information such as image, audio, video files; it also considers internet search indexing, genetic information related datasets, medical data, social network oriented data and log records which have been generated by sensor nodes(Shen, Li, Wu, Liu, & Wen, 2013) (Chen, Mao, & Liu, 2014) (Blondel , Decuyper, & Krings, 2015)The above mentioned types of data are usually unstructured and semi structured. It is not possible to process and maintain the data using existing relational databases as relational database architecture consists of a fixed data field. Insertion of heterogeneous data into existing databases complicates the situation. Thus the Hadoop framework introduces a distributed non-relational database system where huge amounts of data can be stored . HBase is fully column-oriented tabular structure where only structured data e.g. XLS table, MySQL table etc (M. Islam, 2014) can be stored but in our proposal the existing HBase table requires to insert structured, un-structured and semi-structured data respectively. Thus some further efficient and effective preprocessing mechanisms are required for the conversion of unstructured and semi-structured data into structured format to handle the heterogeneity of a large set of data(Nestorov, S., Abiteboul, S., & Motwani,1997). Few authors like Liu and Shi (2015) who have proposed a framework wherein Business Data is analyzed despite the various challenges like quality issues, overall size and heterogeneity along with cross referencing with public domains.

There are various researchers like (Piratla and Jayasumana,2008) who have also introduced column-oriented database systems which later on gained the attention of many researchers working on BigData Analytics . In the column oriented database concept which includes Column-stores, in a nutshell it stores each database table column separately, with attribute values belonging to the same column stored contiguously, compressed, and densely packed, as opposed to traditional database systems that store entire records (rows) one after the other. However, the conversion of unstructured data and semi-structured data is not much focused on. Therefore, the proposed system highlights a technique that can address this issue. The paper is organized as follows. Section I introduces the reader into the various data types which have to be handled by Big Data. The next section summarizes the existing techniques which talk about the Big Data management using HBase followed by an overview of the problem and the proposed system respectively . After that , we give the comparative analysis of the proposed system with respect to the existing Data Transform Method and finally summarize the whole paper.

RELATED WORK

This section discusses about the existing studies towards managing large databases. Wassan,2016 has discussed the importance of Big Data in Health Care and the various challenges associated with it like heterogeneity. Abdelhafez, (2014) has focused on the challenges of Big Data namely variety, volume and velocity and how the current technologies can be used to mitigate these challenges.

Abadi ,Marcus, Madden, and Hollenbach (2007) showed how multi-row distributed transactions with Global Snapshot Isolation (GSI) could be easily supported by using bare-bones HBase with its default configuration so that the high throughput, scalability, fault tolerance, access transparency and easy deployability properties of HBase could be inherited (Abadi et al., 2014). In this paper, the authors have designed a novel and cost effective approach to use HBase as a cloud database solution. A scalable Resource Description Framework (RDF) store based on HBase, which was distributed and column oriented database had been modeled by (Zhang & Sterck, 2010) it could have been utilized for storing RDF triples into six HBase tables. Kaoudi & Koubarakis (2013) and Daneshyar & Razmjoo (2012) have used Hadoop system to build the cloud computing environment. They have used data hiding technology to embed data into cover images, the study also showed that the approach using cloud computing would take less execution time than using a single computer when processing a huge amount of data thus, cloud computing had provided a convenient platform and also reduced the cost of the equipment required for processing gigantic amounts of data. W. Jiang et al . introduced a concept of new storage system which had been termed as Virtual Experiment Storage Service (VESS) for unstructured data from Multi-Disciplined Virtual Experiment Platform (MDVEP) which could improve the performance of the system and simplify the development of upper-layer applications (Jiang, 2011). It provided HTTP-based data request method as system user-friendly interface. Yang, Liu, Hsu, Lu, & Chu (2013) have used HBase, a non-relational database, to further integrate data. The purpose of this paper was to construct complete import tools and solutions based on HBase to facilitate easy access of data in HBase. Apart from it, a visual interface also had been considered to manage HBase to implement user friendly client connection tools for the (Yang et al., 2013) HBase database. The proposed system used the concept of and enhanced the Data Transform Method into No-SQL database or HBase with respect to three different algorithmic forms which are 1) Centralized structured data to Distributed Structured data, 2) Unstructured to structured format and 3) Semi structured to structured format respectively. The performance metrics of the proposed mechanism has been evaluated in a real time scenario and the comparative analysis of the proposed framework performed with respect to the contribution of Yang et al. (2013) where the performance parameters were considered as memory consumption and request per write respectively. The proposed algorithm aimed at providing a framework for handling heterogeneous Big Data .

A sample modeling of general framework, based on extracting patterns and generating recommendations had been reviewed for maintaining and accessing Electronic Health Records (EHRs), which was a form of Big Data and was earlier introduced by Wassan & Jyotsna Talreja (2014). The main focus of the paper was to propose a heterogeneous data handling tool and data storage to store and analyze data with MapReduce paradigm. It was performed on simulated healthcare data. Rallapalli, Sreekanth, & Gondkar (2015) proposed a method which could analyze the Elastic MapReduce on cloud. Hive was used to analyze large data of healthcare and medical records. Sqoop was used for easy data import and export from structured data stores such as relational databases, enterprise data warehouses and No-SQL systems. Apache Hadoop had emerged as a software framework for distributed processing of large datasets across large cluster of computers. Hadoop was based on simple programming model called MapReduce. Hive was a data warehousing framework built on top of Hadoop and also designed to enable easy data summarization, adhoc querying and analysis of large volume of data.

Sahane, Manisha, Sanjay, & Razaullah (2015) studied Hadoop and associated technologies with focus on Map Reduce and analyzed university research data sets to know the focused area of research in Zoology and Botany department. Shao and Conrad, (2015) aimed to quantitatively compare the latencies of different data stores on storing and querying proteomics datasets. They had also introduced the benchmark studies for typical relational and non-relational systems for both, in-memory and disk-based configurations and compared them with a simple flat-file based approach. They also focused on the latencies of storing and querying proteomics mass spectrometry datasets and the actual space consumption inside the data stores. Experiments were carried out on a local desktop with medium-sized data, which were the typical experimental settings of individual bioinformatics researchers. Results showed that there were significant latency differences among the considered data stores (up to 30 folds). In certain use cases, flat file system could have achieved comparable performance with the data stores.

Padhy, Rabi, Manas, & Suresh (2011) proposed a study to provide a better understanding of the non-relational database design, architecture and comparison between them and also identify important research directions in this increasingly important area. The study of Bhagat and Amit(2015) highlighted useful information for companies or organizations using richer and deeper insights and getting an advantage over the competition. For the above mentioned reason, Big Data implementations were needed to analyze and execute as accurately as possible. Due to the rapid growth of such data, solutions need to be studied and provided in order to handle and extract value and knowledge from these datasets. Furthermore, decision makers should be able to gain valuable insights from such varied and rapidly changing data, ranging from daily transactions to customer interactions and social network data. It was found that such value could be provided using Big Data analytics, an application of advanced analytical techniques on Big Data. In order to process these large amounts of data in an inexpensive and efficient way, parallelism was used.

Pokorný & Jaroslav (2014) focused on No-SQL databases which support solving, at least partially, Big Data problems. Some features of No-SQL databases like data models and querying capabilities had been presented in more detail. They have also mentioned an overview of some their representatives. Finally, they pointed out the actual problems associated with current database research.

Woo, Jongwook, & Xu (2011) presented Market Basket Analysis algorithm with MapReduce, one of popular data mining algorithms. The algorithm had been used to sort data set and to convert it to (key, value) pair to fit with MapReduce. The experimental prototype executed on Amazon EC2 Map/Reduce platform. The experimental result has shown that the code with MapReduce increases the performance by adding more nodes but at a certain point, there was a bottle-neck that does not allow the performance gain. It was believed that the operations of distributing, aggregating, and reducing data in MapReduce should cause the bottle-neck. Duggal, Puneet, & Sanchita (2013) suggested various methods for catering to the problems in hand through Map Reduce framework over Hadoop Distributed File System (HDFS). MapReduce techniques have been studied in this paper which is implemented for Big Data analysis using HDFS.

PROBLEM DESCRIPTION

The current scenario in the field of healthcare systems requires managing massively growing Big Data sets either in a local repository of an individual healthcare center, a chain of many healthcare centers of a particular region or at national level, i. e collaboration of these centers. It can be seen that there is very less amount of computing resources in a medical health care system to process a huge amount of heterogeneous data in a parallel and distributed fashion. In the human life cycle right from the child in a womb till his/her death all the clinical, pathological, radiological and advance investigations should be maintained in a distributed manner so that it ensures the ease of accessibility of those files in future e.g. in case of medical claims, further proceedings on biomedical research and applications. Storing, maintaining and analyzing those files throughout a human life cycle will reduce the cost of analysis and delay in treatment. The investigations from previous diagnosis also increase the accuracy of treatment of a specific disease. In recent times, storing and processing of heterogeneous BigData and analytics of BigData have become very challenging especially in the health care sector. The existing research trends talk about only the structured data and how the structured data can be stored into HBase in a No-SQL manner but there are no existing studies which talk about the storing, managing and analysis of heterogeneous data such as (semi-structured , unstructured data) in the HBase which is a non-relational distributed data base with higher implementing feasibilities. The current situation also needs an efficient and collaborative data model which can be applicable in different layers of data structure and processing such as data acquisition, data storage , data search and data analysis respectively. There can be un-structured and semi structured data which cannot easily proceed with existing traditional databases such as My SQL, Oracle etc. Another challenging situation in the field of relational database management systems include lack of parallel and distributed computing scenarios for managing the heterogeneity of data for example (HBase needs to store a patients X-ray report, MRI scan report which are unstructured in nature . It also needs to store semi structured data such as XML files). An efficient analytical prototype has to be developed for optimizing the processing time with the context of parallel and distributed computing.

PROPOSED SYSTEM

This section gives an overview about the proposed system which basically handles three different types of datasets namely structured, semi-structured and unstructured format of the data. There is a minor limitation associated with the HBase data handling mechanisms as it only can store the data in a structured format. To optimize the heterogeneity of data, an algorithm has been introduced which includes three different types of procedures 1) Unstructured to structured format, 2) Centralized structured data to distributed structured data and 3) Semi-structured to Structured format. This study focuses on building a data handling tool which performs preprocessing of semi-structured and unstructured data format for the ease of storing into HBase table, thereby handling heterogeneity.

The first procedure discusses about how unstructured data can be stored into HBase. In algorithm one, we consider text files where patient information will be present in an unstructured format and then it performs knowledge based extraction from the patient details and inserts the values into HBase in a structured manner using Map Reduce function. The Map Reduce function generates the master list and converts the master list into list of puts. After that it puts all the values into HBase. There are two different modules that have been discussed in the above mentioned procedure where an administration and department module framework have been designed and evaluated.

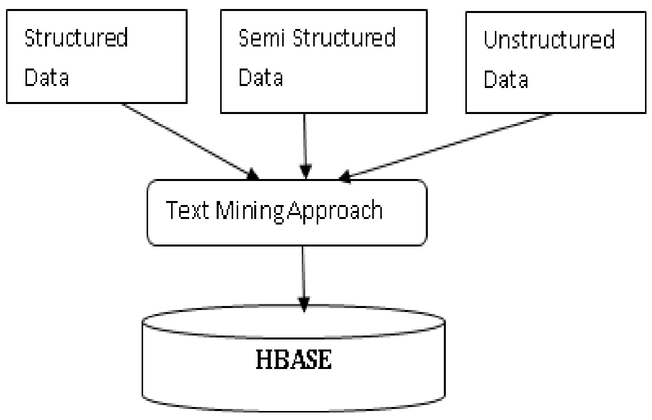

Figure 1 Architecture of the Proposed System

These modules introduce clustering mechanisms where different distributed directories belong to different hard disk drives and their respective directories are considered as clusters and a cluster can consist of n number of text files where the patient details along with doctor note will be present in an unstructured format. The preprocessing of the unstructured data includes a data mining technique as in Figure 1 where a filter has been added for extracting meaningful information from the patient query details. MapReduce is considered here as data mining procedure for the conversion of unstructured data into structured format. In the first procedure the unstructured data format has been converted into semi-structured format and then it is pushed into HBase. It also manages directories of distributed clusters. The technique evaluates how existing traditional structured data format can be stored into HBase in a No-SQL fashion.

The prime contribution of the proposed study is to develop a data conversion framework for handling complex heterogeneous health care Big Data. The proposed Technique considers three different types of data conversion procedures namely 1. Conversion of Unstructured Data to structured Format 2. Conversion of Conventional Structured data to Distributed Structured Data and 3. Conversion of Semi Structured data to Distributed Structured format. This section introduces a proposed text mining approach which ensures very less amount of computational complexities as well as processing time. Here the data set is a set of clinical notes which is unstructured in text files. The performance metrics associated with the proposed system is highlighted pictorially in Section VII. The unstructured clinical notes are inserted into HBase using Map Reduce. There are six preprocessing steps in conversion of Unstructured data to Structured Format.

1) Reading data into the program

2) Identifying Category i.e Document Annotation

3) Separation into new lines

4) Token Identification

5) XML Generation

6) Generation of key values which are pushed into HBase

A. Unstructured Clinical Note (Set of text files) →HBase

The purpose of the following algorithm is to insert unstructured clinical notes which are usually considered as a set of text files into Hbase. The following algorithm converts list of maps into list of puts.

Algorithm for Unstructured Clinical Note to HBase

```
Input: $M_{list}$
Output: $P_{list}$
Start
1. Generate Masterlist $\leftarrow$ Read data reposit
2. Map Reduce operations $\leftarrow$ Batch processing for parallelism
3. for ($m \in M$)
4.  Retrieve $\leftarrow R_{k \in m}$  && Create $\rightarrow P_v$
5. (Nested for each $R_k \in m$)
7. Put $\xleftarrow{Add}$ $R_k$, V
```

```
8. End Nested for
9. List $\leftarrow P_v$
10. End for
11. $H_B \leftarrow P_v$
End
```

The above algorithm takes M_{list} which is stated as list of Maps and generates P_{list} which is list of puts. Put means putting into the map and then storing in HBase. Here M, m, R_k ,P_v and H_B have been used to denote a set of maps , a single map , row key , put value , single value and HBase respectively.

B. ***Generate Masterlist***

The purpose of Generate Masterlist is to perform data mining and classification among the clinical notes where files have been annotated with appropriate keyword lists and the framework generates annotated documents. The reason for annotation is that later when we try to retrieve records, through these annotated words retrieval becomes fast.

Algorithm for generation of Masterlist

Input: F_n , K_l

Output: Annotated with appropriate K_w

```
Start
1. Initialize $\leftarrow f_l$
2. Initialize $\leftarrow K_{list}$
3. for (i = 1 to n)
4.Read $\leftarrow$ each $f \in D_i$
5. Annotate $\leftarrow f_i \prod K_{list}$
6. Read $\rightarrow f_i$
7. for ( i=0 to n)
8. if ($K_w \in K_{list}$ && $K_w \in f_i$)
9. $A_{list} \leftarrow ADD_{k_w}$
10. End of if
11. End of for
```

```
12. Map $\xleftarrow{Put} A_{list}(K_W)$
```

End

The above mentioned algorithm takes F_n and K_l which are denoted as file name and keyword lists as input and generates annotated keyword lists K_W. The keywords f_i, D_i are stated as Files and their respective directory. A_{list} signifies the list of arrays which uses keyword list and put for generating Map. After this procedure, K_{list} will be a list of all keywords of diseases

C. *Identify the Sentence Boundary*

The purpose of identifying the Sentence boundary is to convert the lines of data which belongs to a particular file into list of key and value pairs. The proposed algorithm is defined below.

Algorithm for identifying the Sentence Boundary

Input: $L_D \in f_i$

Output: List (Key, Value)

Start

```
1. for (i= 1 to n)
2. Select $\rightarrow L_{D_i}$
3. $S_D \xleftarrow{Split(Tab|Space)} D_i$
4. Tokenize
5. List $\xleftarrow{AddTokens} T_i$
6. End for
```

End

The above algorithm takes lines of data L_D as input and produces a list of key and value pairs as output. The above mentioned algorithm selects lines of data and split them on colon, tab and space. The split data (S_D) are tokenized (T_i) and added to the list.

D. *Tokenization of each word*

Tokenization is performed to identify individual tokens in each line. It also takes lines of data as input, generate tokens and add them to the token lists. Tokenization also eliminates stopwords.

Algorithm for tokenization of each word

```
Input: $L_D \in f_i$
Output: List ($T_i \in L_i$)
Start
1. For (i=0 to n)
2. Select $\leftarrow$ each $L_i$
3. $L_T \xleftarrow{Tokenize(Colon|Tab|Space)} L_i$
4. $T_{list} \xleftarrow{Add} T_{k_i}$
5. End for
End
```

E. ***Generate XML***

XML files are generated from row keys where it takes key value per line as input and generate XML format of that key values.

Algorithm for Generating XML

```
Input: $K_v \in L_i$
Output: XML Format of $K_v \in L_i$
1. for (i= 0 to n)
2.  Select $\leftarrow L_i$
3. Split on colon $\leftarrow L_i$
4. for ( i= 0 to n)
5. if ( $N_T > 1$ && $\sum T_i \in K_i$ )
6.  Do $\leftarrow$ Concatenation Operation with under score
7. End if
8. Generate $\leftarrow$ XML
9.  End for
End
```

In the above algorithm K_v, L_i have been denoted as key value per line and lines respectively. N_T, T_i stands for number of tokens and tokenized keyword. The algorithm generates an XML file using a concatenation operation with under score.

*F. **Create Map***

The purpose of creating Map is to load all the key values about an object into Map. The following algorithm takes list of keywords as input and creates a Map.

Algorithm for Creating Map

```
Input: $L_k$
Output: M
Start
1. for (i=0 to n)
2. Select $\leftarrow L_i$
3. if ( $: \in L_i$ == True)
4. Colon $\leftarrow$ Split
5. M $\xleftarrow{Put}$ $K_V$
6. End if
7. End for
8. Master list $\xleftarrow{Add}$ M
End
```

In the above mentioned algorithm notations used are M, K_v, and Li which denotes Map, Key value per lines and Lines respectively. The algorithm takes list of keyword (L_k) as input and generates a Map which further added to a Master List.

*G. **Add Map to Master list***

The purpose of adding Map to Master list is to apply the user preferences to the search. The following algorithm takes requirements of users as input and in output the algorithm applies the selected user preferences to a middle ware map reduce program dynamically.

Algorithm for Add Map to Master list

```
Input: $R_U$// Requirement of Users
Output: User choices applied to middleware Map Reduce program dynamically.
Start
1. Search $\leftarrow$ FilterRowedFile
2. Search $\xleftarrow{U_p()}$ Get ( $U_p$ )//User Preferences
```

```
3. $E_p \xleftarrow{Add()}$ Filter // Each Preferences
4. for (i=0 to N)
5. Select $\leftarrow U_{C_i}$ // User Choices
6. Create $\leftarrow F_i$ // Filter
7. $F_L \xleftarrow{Add} U_{C_i}$
8. End for
9. Apply $\leftarrow F_L$
10. Scan $\leftarrow H_B$
End
```

The above mentioned algorithm uses the notations as R_U , U_P, E_P, U_C , F_i , F_L and H_B which stands for user requirements , user preferences , each preferences , user choices , Filter, Filtered List , HBase. The proposed algorithm takes the user choices and filters it. The filtered list further is processed to HBase.

H. ***Retrieve from HBase***

The purpose of this technique is to automatically filter records based on department module. The proposed technique takes user department as input and displays the output as filtered record.

Algorithm for Retrieve from HBase

```
Input: $U_D$ // User Dept
Output: $F_R$ ( $R_c, K_i, Link$ ) // Filtered record
Start
1. for (i=0 to N)
2. Select $\leftarrow R_i$
3. for( i=0 to N)
4. Select $\leftarrow D_i$ // Department in case of Admin
5. if ($D_i \in A_D$) // Annotated Doc
6. Count ++
7. Create $\leftarrow$M
```

```
8. Path $\xleftarrow{Get(Row_K)}$ Put
9. $R_L \xleftarrow{Add} M$ //Map //Result List
10. End if
11. End for
12. End for
13. Access to the Doc in read only format.
End.
```

The proposed algorithm takes U_D (User Department) as input and generates filtered records (F_R). In the algorithm D_i, A_D stands for department in case of Admin and annotated document. The algorithm also creates Map (M) and put it to the appropriate Path using the function get (Row_k), finally the Map is put to the result list and a user can view the document in a read only format.

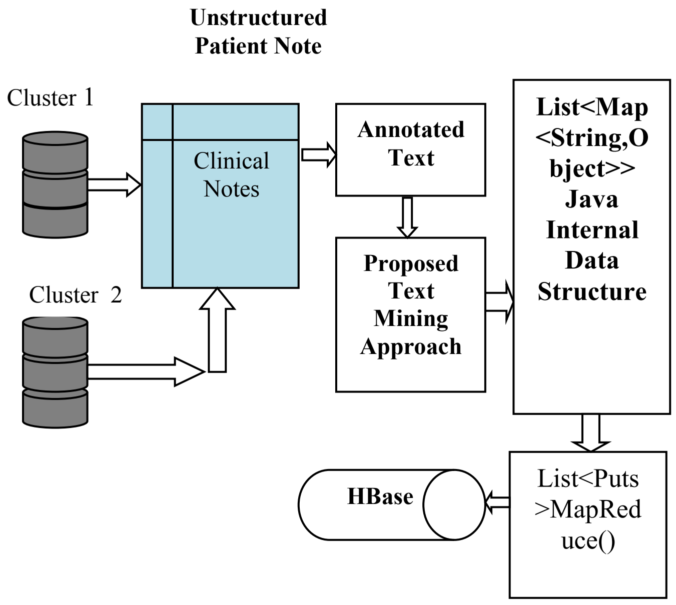
Unstructured Patient Note
Cluster 1
Cluster 2
Clinical Notes
Annotated Text
Proposed Text Mining Approach
List<Map <String,Object>> Java Internal Data Structure
List<Puts >MapReduce()
HBase

Figure 2 Unstructured to Structured Data Format

Figure. 2 shows the implementation details of the proposed algorithm one which is unstructured to structured data format.

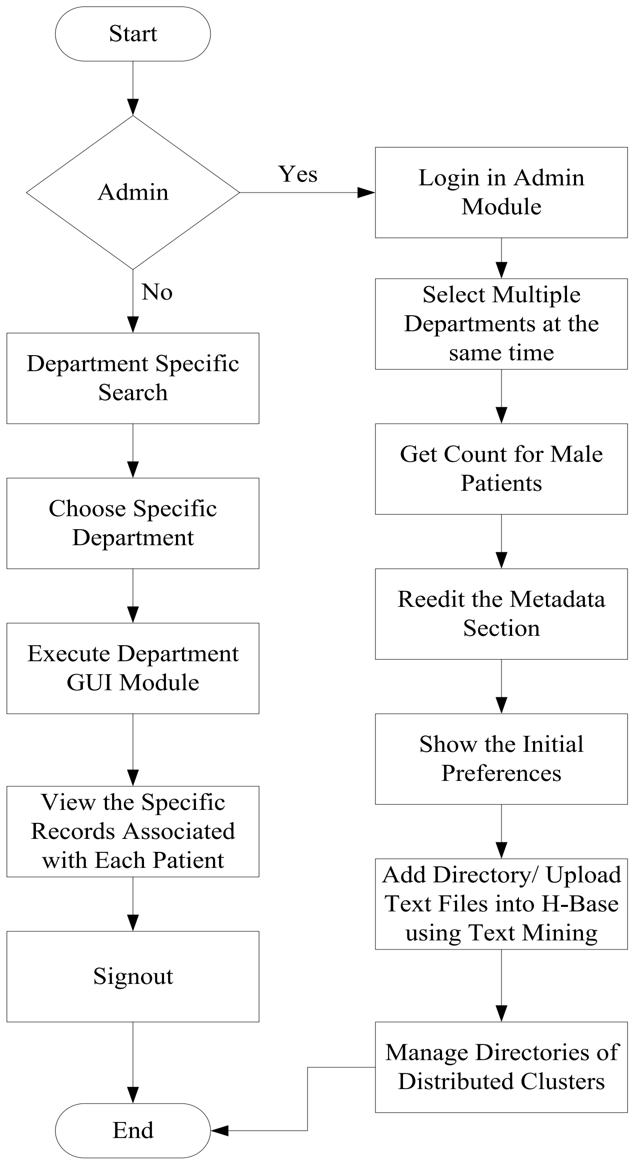
Start
Admin
Yes
No
Login in Admin Module
Select Multiple Departments at the same time
Get Count for Male Patients
Reedit the Metadata Section
Show the Initial Preferences
Add Directory/ Upload Text Files into H-Base using Text Mining
Manage Directories of Distributed Clusters
Department Specific Search
Choose Specific Department
Execute Department GUI Module
View the Specific Records Associated with Each Patient
Signout
End

Figure 3 Flowchart of Unstructured Data to structured data format

The flowchart associated with the patient information retrieval is discussed in the flowchart given.

The department module shows the step by step procedure of how an unstructured data will be stored and retrieved from an HBase table. Here first of all the query should be generated to get the specific patient details of some mentioned years and query should be filtered to retrieve precisely the number of records associated with the search item e.g. patient name and its details. As per the query, the module will execute and generate the patient details which are maintained by different text files placed in different clusters. Another module which is termed as Admin module for the proposed algorithm has been introduced below.

The next procedure is to convert Centralized structured data to Distributed Structured data. It also evaluates the insertion/storing of structured data (Here the patient information, family history and prescription have which been considered as structured data) into an HBase will be performed with respect to multiple columns associated with a single row key. Some constraints have been raised for optimizing the size of row and column and the proposed algorithm also ensures low memory consumption in terms of data loading into HBase. Figure 4 represents the graphical representation of the algorithm two which represents the conversion of conventional structured data (XLS spread sheets) into structured format of HBase.

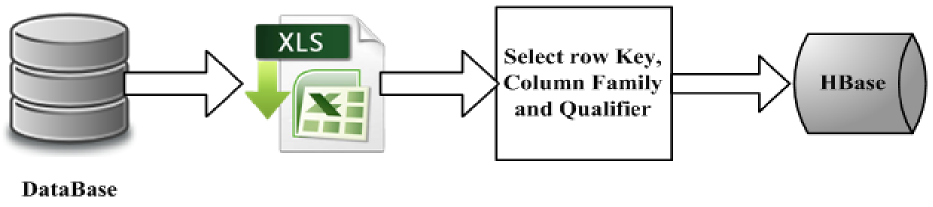

Figure 4 Centralized structured data to Distributed Structured data

The proposed framework reads the XLS spread sheet which contains the patient information, then reads the file from a cluster or directory, later XLS file is analyzed with respect to various row keys, column names and column qualifiers. We have used the POI(Poor obfuscated Implementation) Technology since it has to be converted to a format which the HBase can handle. A middle-tier mapping functionality maps the different types of values associated with each patient into a single data type as HDFS and HBase tabular architecture only support one particular data type and it has got its particular internal architecture. Hadoop file systems and its internal database architecture stores the structured values in a different format. After considering the row key and its associated values the proposed algorithm inserts the values into the HBase.

Figure 5 highlights the process of how structured data can be loaded or inserted into an HBase Table from a XLS file. The algorithm takes the input as a structured table which can be XLS or MySQL tables. The proposed algorithm for conversion of Centralized structured data to Distributed Structured data also highlights how the row keys are initialized and mapped with their respective cells where some data can be present. It also searches for the data of each cell associated with a specific row key and take it out from that table for storing it into an HBase table. In the structured or existing traditional databases the values can be of some specific data types but the proposed system, it first read the data from the structured database then it analyzes and determines its data type and map it into a particular variable irrespective of data types.

Algorithm for Centralized structured data to Distributed Structured data.

```
Input : $t_i$ where $t_i \in T$ and $i \in R_1^n$
Output: $H_{t_i}$
Start
1. For each $t_i$
2. Do $T \leftarrow t_i$
3. for ( $i \leftarrow 0$ to $n$)

4. Count $\leftarrow r_i$
5. Initialize $\leftarrow t_{i_{r_i}}$
6. Nested for ($i \leftarrow 0$ to $n$)
7. Search $\leftarrow t_{i_{r_i}} C_i$
8. If ($C_i \in Data$)
9. $H_{t_i} \leftarrow Data$
10. Else
11. Ignore
12. Continue $\rightarrow$ Iteration
13. End
```

In the above algorithm t_i is taken as input (where $t_i \in T$ and $i \in R_1^n$) and the output generated is H_{t_i}. Parameters like t_i , T , i , H_{t_i} , r_i , $t_{i_{r_i}}$ and C_i are considered as Denotes Table which will be taken as input, A set of structured Table (e.g. XLS , My SQL) , A set of real numbers, H-Base Table , A set of Rows , Row key associated with specific table and cell associated with each row key respectively.

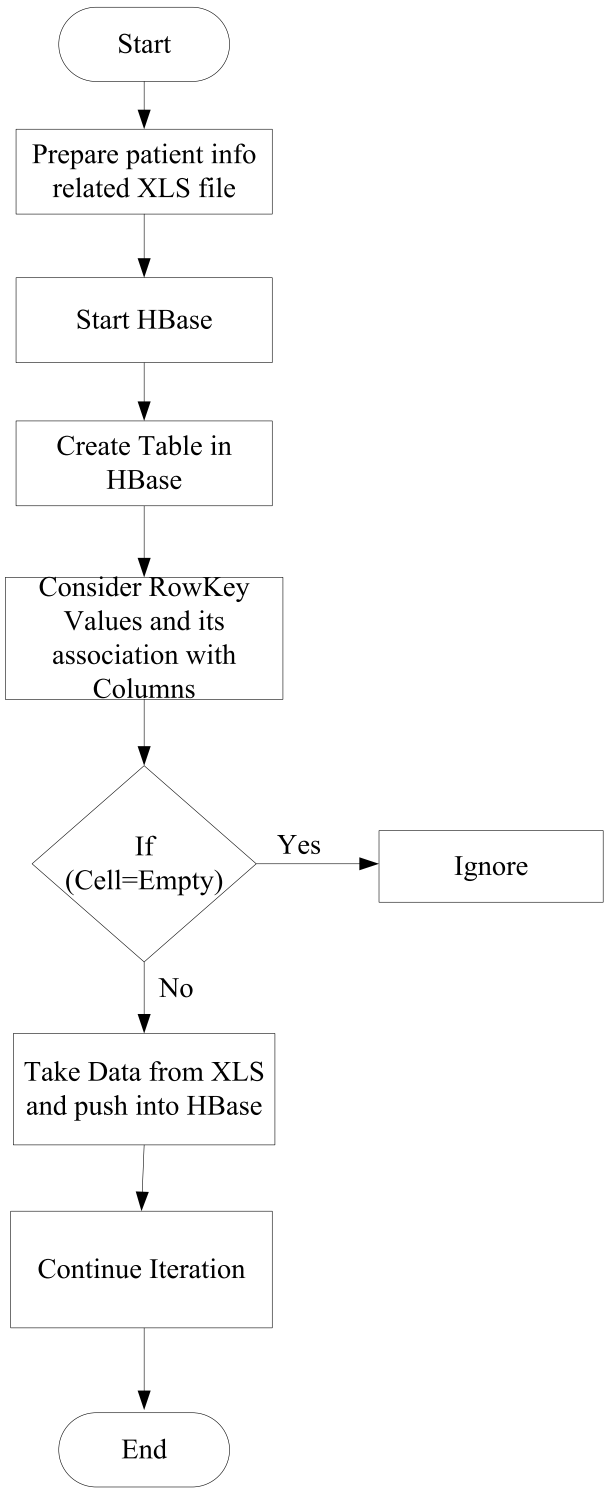
Start
Prepare patient info related XLS file
Start HBase
Create Table in HBase
Consider RowKey Values and its association with Columns
If (Cell=Empty)
Yes
Ignore
No
Take Data from XLS and push into HBase
Continue Iteration
End

Figure 5 Flowchart of Centralized structured data to Distributed Structured data

The algorithm for Centralized structured data to Distributed Structured data discusses about the process of how semi structured data can be stored into HBase. The proposed algorithm utilizes a Java parser which performs the parsing of any semi structured file here. The final phase of our approach is to convert Semi-structured Data into Structured format.

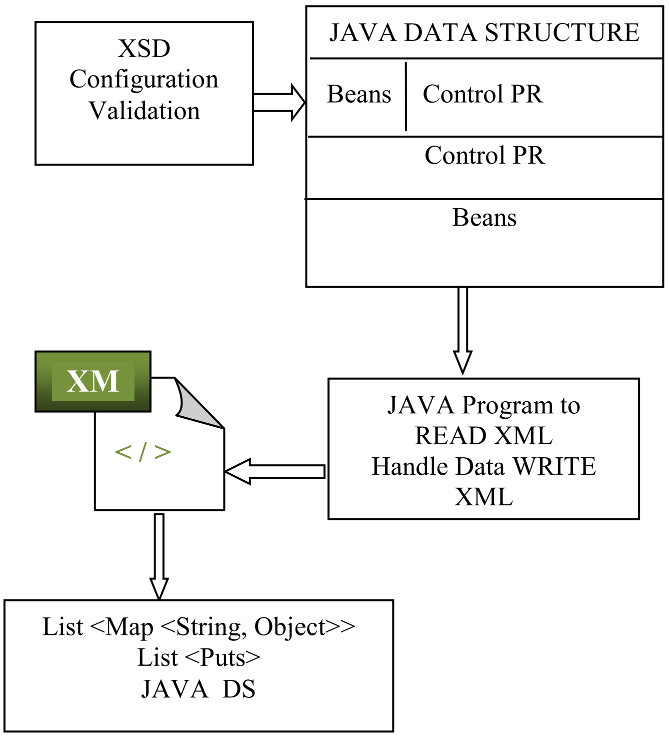
XSD Configuration Validation
JAVA DATA STRUCTURE
Beans
Control PR
Control PR
Beans
XM
</>
JAVA Program to READ XML Handle Data WRITE XML
List <Map <String, Object>>
List <Puts>
JAVA DS

Figure 6 Semi structured data to Distributed Structured data

Figure 6 shows how semi structured data can be stored into HBase in a structured form. The process of the proposed system considers a schema structure which is a XML format or XSD file. It converts the XSD structured data into a different data structure maintained by java classes. It uses three different types of procedures which are

1. Parsing the schema
2. Compiling the schema
3. Generation of some java classes

The generation of java classes also includes some sub procedures which are

- Playing java bin classes $\rightarrow$ similar to the struct in C
- Generation of Patient list module
- Patient type module

One of the programming module has been evaluated using the above mentioned data structure for generating a java specific implementation and representation of the XML schema. There is a converting tool which converts XSD file to java specific implementation and generate a control program for handling other modules. The following Figure7 highlights the process flow of the proposed algorithm .

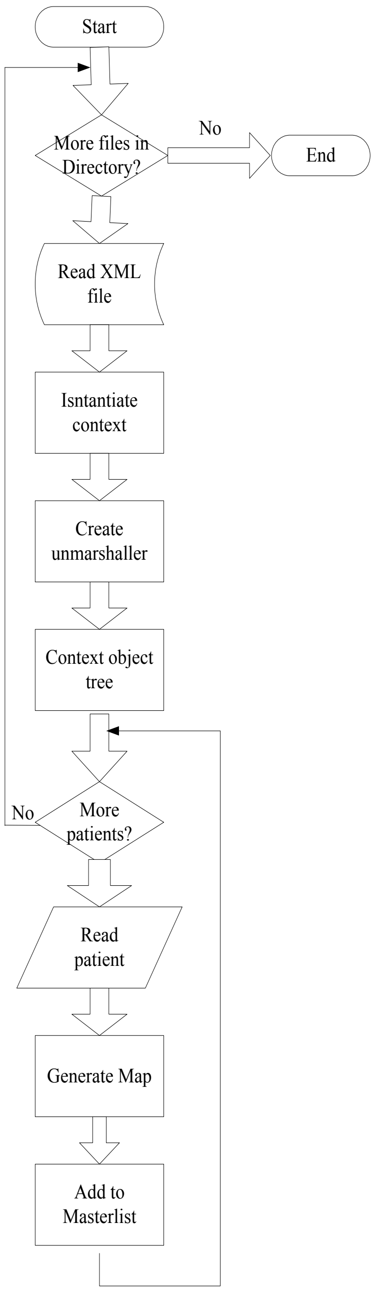

Figure 7 Flowchart of Semi Structured to Structured Format

Algorithm For Semi structured Data to Structured Format

Semi-structured => HBase

=========================

XML => Middle tier => HBase

XML => Middle tier

===================

Input: XSD

Output: i) ADT

ii) Top-level program that consumes the ADT

1. Conversion tool generates the schema equivalent programs in the middle tier

XSD $\xrightarrow{ConvertionTool}$ Middle tier programs

input: .xml

output: List of Patients

2. Write a program that unmarshall the xml to generate an in-memory object unmarshall

xml file/ directory ==========> Object

Algorithm:

```
For each xml file, that contain patient(s):
        * Instantiate the context
        * Create unmarshaller
        * Produce a tree of content objects, equivalent to xml file
        for each patient:
                * Generate a map representation of all the patient properties and attributes
                * Add map to the master list
```

The above algorithm shows how semi structured data can be converted into distributed structured data. The whole process includes middle tier programs which have been generated using a conversion tool. The algorithm takes .xml format as input datasets and generates a map representation of all the patient related properties and attributes. The process also includes creation of an unmarshaller file, a tree structure of content objects and in the end it adds the created Map to the Masterlist. Unmarshaller is a class which is used to create tree like structure for handling XML tree.

RESULT ANALYSIS

The experimental prototyping has been evaluated to compare the proposed system with the existing Data Transform Method (DTM) for handling data heterogeneity in the field of Big Data analytics.

Table 1 Cost of different configuring properties of HBase

HBase Properties/Configuration	A	B	C	D	E
setAutoFlush()	On	On	On	Off	off
setWriteBufferSize(10TB)	Off	On	On	On	On
setWriteToWAL()	On	On	Off	On	Off
Time (Sec)	935	918	634	50	45

Table 1 and Table 2 shows the time required to put the records into HBase by setting three parameters setAutoFlush(), setWriteBufferSize() and setWritetoWAL(). It can be seen that the proposed Unstructured to Structured algorithm consumes very less amount of time during configuration as compared to the existing DTM methodology even under varying data sizes.

Table 2 Set HBase Region server Handler Count Value to 20

HBase Properties/Configuration	A	B	C	D	E
setAutoFlush()	On	On	On	Off	off
setWriteBufferSize(8TB)	Off	On	On	On	On
setWriteToWAL()	On	On	Off	On	Off
Time (Sec)	905	894	600	53	39

The following graphical analysis also highlights the effectiveness of the proposed system. The experimental prototype for write request/sec has been evaluated in Hbase configuration .

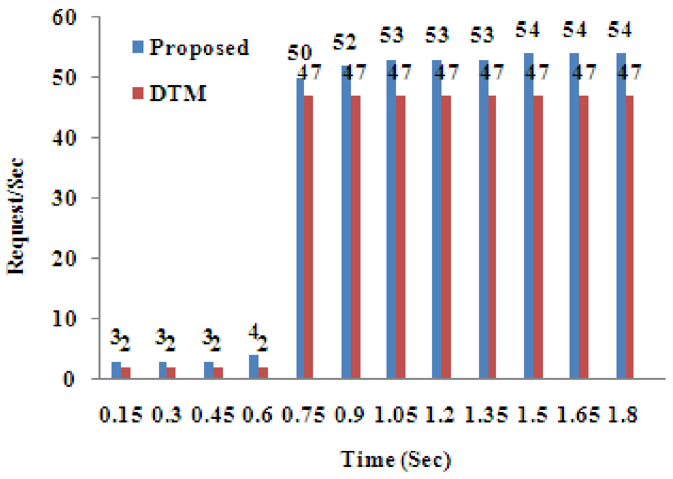

Figure 8 Comparative analysis of the proposed system with respect to DTM

Figure 8 shows that proposed system uses a very less amount of JVM heap committed memory while executing the codes for data requests, when compared with the existing DTM technique where it is quite higher.

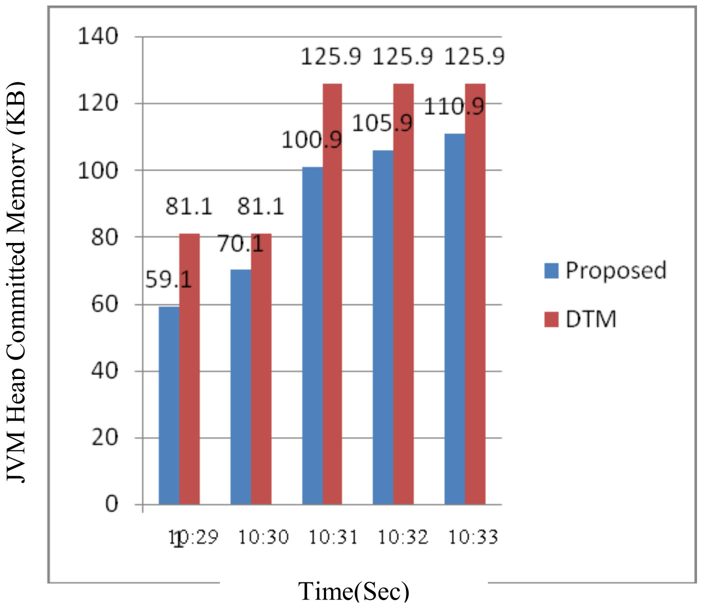

Figure 9 JVM heap committed memory (KB) vs. Time (Sec)

It shows that the algorithms are computationally much efficient with very less amount of computational complexity. The experimental test bed has been executed in a real time scenario for different modules for execution of the complete program and analyzed properly. It can be said that the proposed framework can be utilized in future research direction of Big Data Analytics. During experimental prototyping the proposed system consumes very less amount of JVM heap committed memory with respect to time as highlighted in Figure 9. It can be seen in Figure 7 that it also performs better than the existing DTM technique while evaluating amount of requests with respect to time. The comparative analysis shows that the proposed system ensures higher cost effectiveness and efficiency as compared to existing DTM technique.

CONCLUSION

Managing heterogeneous health care Big Data in a real-time and distributed scenario has so many constraints and hence there is an urgent need for developing an efficient Data Handling tool. This study introduces a data conversion model for handling complex heterogeneous health care Big Data using Hadoop oriented distributed computing architecture. In this study, a cost effective and computationally efficient text mining method has been introduced for conversion of heterogeneous Big Data into a distributed structured data . The uniqueness of this proposed study is that the proposed work has been carried out using a real time experimental prototyping. Hence the performance metrics of the proposed system also ensures usability of the proposed healthcare Big Data handling tool in future research domains related to handling of heterogeneous medical Big Data. The result analysis section represents the comparative analysis of the proposed approach versus the existing Data Transform Method (DTM). The graphical analysis also highlights the efficiency and the effectiveness of the proposed system.

ACKNOWLEDGEMENT

We would like to extend our gratitude to the members of PRO-ACT who have developed the data set used in this work. PRO-ACT (Pooled Resource Open-Access ALS Clinical Trials Database) contains about 8500 records of ALS patients whose identity is hidden. In 2011, Prize4Life, in collaboration with the Northeast ALS Consortium, and with funding from the ALS Therapy Alliance, formed the Pooled Resource Open-Access ALS Clinical Trials (PRO-ACT) Consortium. The data available in the PRO-ACT Database has been volunteered by PRO-ACT Consortium members. (https:nctu.partners.org)

REFERENCES

[1] Abdelhafez, H. A. (2014). Big Data technologies and analytics: A review of emerging solutions. *International Journal of Business Analytics (IJBAN)*, *1*(2), 1-17.

[2] Abadi, D. J., Marcus, A., Madden, S. R., & Hollenbach, K. (2007, September), Scalable semantic web data management using vertical partitioning. *Proceedings of the 33rd International Conference on Very Large Data bases*, 411-422.

[3] Assunção, M. D., Calheiros, R. N., Bianchi, S., Netto, M. A ., & Buyya, R. (2015). Big Data computing and clouds: Trends and future directions. *Journal of Parallel and Distributed Computing*, *79*, 3-15.

[4] Bhagat, A. (2015). Understanding Big Data: Framework and tools for massive data storage and mining. *International Journal of Engineering and Technical Research (IJETR), 3*(6), 305-308.

[5] Bihl, T. J., Young II, W. A ., & Weckman, G. R. (2016). Defining, understanding, and addressing Big Data. *International Journal of Business Analytics (IJBAN*), *3*(2), 1-32.

[6] Blondel, V. D., Decuyper, A ., & Krings, G. (2015). A survey of results on mobile phone datasets analysis. *EPJ Data Science*, *4*(1), 1.

[7] Chen, M., Mao, S., & Liu. Y. (2014). Big Data: A survey. *Mobile Networks and Applications*, *19*(2), 171-209.

[8] Daneshyar, S ., & Razmjoo, M. (2012). Large scale data processing using mapreduce in cloud computing environment. *International Journal on Web Service Computing* (IJWSC), *3*(4), 1-13.

[9] Duggal, P. S., & Paul, S. (2013, November). Big Data analysis: Challenges and solutions. In *International Conference on Cloud, Big Data and Trust,* 13-15.

[10] Islam, M. (2014) . A cloud based platform for Big Data science. *DiVA Portal.*

[11] Jiang, W., Li, H., Jin, H., Zhang, L., & Peng, Y. (2011). VESS: An unstructured data-oriented storage system for multi-disciplined virtual experiment platform. *In Proceedings of the International Conference on Human-centric Computing and Embedded and Multimedia Computing*, 187-198.

[12] Kaoudi, Z ., & Koubarakis, M. (2013). Distributed RDFS reasoning over structured overlay networks. *Journal on Data Semantics*, *2*(4), 189-227

[13] Liu, K., & Shi, J. (2015). A systematic approach for business data analytics with a real case study. *International Journal of Business Analytics (IJBAN)*, *2*(4), 23-44.

[14] Piratla, N. M., & Jayasumana, A. P. (2008). Metrics for packet reordering - A comparative analysis. *International Journal of Communication Systems*, *21*(1), 99-113.

[15] Nestorov, S., Abiteboul, S., & Motwani, R. (1997). Inferring structure in semistructured data. *ACM SIGMOD Record*, *26*(4), 39-43.

[16] Padhy, R. P.,Patra, M. R., & Satapathy, S. C. *(2011)*. RDBMS to NoSQL: Reviewing some next-generation non-relational databases. *International Journal of Advanced Engineering Science and Technologies, 11*(1), *15-30.*

[17] Pokorný ., & Jaroslav. (2013). New database architectures: Steps towards Big Data processing. In Proceedings of the IADIS *International Conference Intelligent Systems and Agents*, 3-10.

[18] Rallapalli, Sreekanth., & Gondkar, R. R. (2015). Map reduce programming for electronic medical records data analysis on cloud using apache hadoop, hive and sqoop. *International Journal of Latest Technology in Engineering, Management & Applied Science, 4*(8), 73-76.

[19] Sahane, M., Sirsat. S., & Khan, R. (2015). Analysis of research data using mapreduce word count algorithm. *International Journal of Advanced Research in Computer and Communication Engineering*, *4*(5), 184-187

[20] Shao, B., & Conrad, T. (2015). Are NoSQL data stores useful for bioinformatics researchers*?, ". International Journal on Recent and Innovation Trends in Computing and Communication, 3*(3), 1704-1708.

[21] Shen, Y., Li, Y., Wu, L., Liu, S ., & Wen, Q., (2013). Big Data overview. *Enabling the New Era of Cloud Computing: Data Security, Transfer, and Management: Data Security, Transfer, and Management*, 156.

[22] Wassan, J. T. (2016). Big Data paradigm for healthcare sector. In *Managing Big Data Integration in the Public Sector* 169-186. IGI Global.

[23] Wassan, J. T. (2014). Modelling stack framework for accessing electronic health records with Big Data needs*. International Journal of Computer Applications*, *106*(1), 37-45.

[24] Woo, J., & Xu, Y. (2011). Market basket analysis algorithm with map/reduce of cloud computing. In The 2011 *International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA 2011),* Las Vegas.

[25] Yang, C. T., Liu, J. C., Hsu, W. H., Lu, H. W., & Chu, W. C. C. (2013, December). Implementation of data transform method into nosql database for healthcare data. In 2013 *International Conference on Parallel and Distributed Computing, Applications and Technologies,* 198-205.

[26] Zhang, C., & De Sterck, H. (2010). Supporting multi-row distributed transactions with global snapshot isolation using bare-bones HBase. *11th IEEE/ACM International Conference Grid Computing (GRID)*, 177-184.

BIOGRAPHY:

C.S. Sindhu is a research scholar at JNIAS, Hyderabad. She was a faculty in the Department of Computer Science for more than a decade teaching undergraduate students. She has a B.E in Computer Science from M.S. Ramaiah Institute of Technology, Bangalore, India and M. Tech in Computer Science from B.V.B. College of Engineering and Technology, Hubli, India. She has around 12 publications of international journals and conferences to her credit. She has presented papers at *IEEE International Conference on Computational Intelligence and Computing Research(ICCIC), ACM International Conference on Big Data & Wireless Technologies, IEEE Sponsored 9th International Conference on Intelligent Systems and Control.* Her work has appeared and will be appearing in *Annals of Information Systems on Big Data Analytics series by Springer, International Journal of Computer Engineering & Technology (IJCET) , International Journal of Innovations & Advancement in Computer Science(IJIACS) and so on.*

Nagaratna P. Hegde, is currently working as Professor at Vasavi College of Engineering, Hyderabad. She has chaired various conferences and is actively trying to propagate Computer Science through her works. She has obtained her Doctorate from NIT, Surathkal. Having 22 years of experience in teaching Computer Science both at undergraduate and post graduate levels, she is presently guiding eight members for their Ph.D. Her research interests include image processing; natural language processing, speech processing, and knowledge based systems, data mining and Big Data analytics and has to her credit around 45 papers published in journals and conference proceedings.

Professor C.S. Sindhu
Research Scholar
Jawaharlal Nehru Institute of Advanced Studies
6th Floor, Buddha Bhavan, M.G Road, Secunderabad - 500 003, Telangana, India
Tel: +91 9886234511
sindhu33in@gmail.com

Professor Nagaratna P. Hegde
Professor, Department of Computer Science & Engineering
Ibrahim Bagh, Hyderabad, Telangana 500031, India
Tel: +91 9108218288
nagaratnaph@gmail.com