2015 Eighth International Conference on Internet Computing for Science and Engineering

Hadoop-based Distributed Computing Algorithms for Healthcare and Clinic Data Processing

Jun Ni, Ying Chen, Jie Sha, and Minghuan Zhang
[1]Big Data and Informatics Research Lab, Shanghai Sanda University
Shanghai, 201209, China
sageli@126.com

Abstract. **There exit a huge demand on utilizing big data technology to process healthcare related patients data for healthcare information extraction and medical knowledge discovery. In this paper, we briefly review the demands and application potentials using big data technology with an emphasis on common challenges. After briefly addressing the Hadoop/MapReduce code components and modules, we use a simple clinic data to demonstrate how to map and reduce on small dataset with illustrated workflow. We give simple scenario of using other MapReduce calculation modules for counting and classification. This serves as a basic step into future utilization of big data to healthcare domain.**

Keywords: **Healthcare information systems, big data, Hadoop, MapReduce, healthcare and clinic data**

I. INTRODUCTION

Level of healthcare systems is used to measure quality of life in today's societies domestically and internationally. Many countries develop national strategy and policies to advocate and promote healthcare systems. For example, the current USA government has been striking to continuously punching the healthcare reformation for whole coverage of healthcare insurance. It has been tough and challenging due to political and economic obstacles. In China, the efforts of improving national level healthcare have also been made aggressively. The achievements are significant and progressive. From Chinese government's white paper, it can be seen that healthcare for whole nation become the one of important strategies based on the demands to improve people's daily life through healthcare systems. The dream is to leverage healthcare so every patient can have appropriate medical treatments and cares, although Chinese healthcare level is below the average in the world and needs to be improved. It is envisioned by the government and 1.4 billion people, representing 19.24% of total world population. Therefore, the government or society operated healthcare systems.

Among many healthcare related missions, the healthcare information systems accelerated by today's rapidly-developed information technology become extremely an important task. Such system plays a crucial role in healthcare related domains such as public health, medical services, healthcare insurance, medicine and pharmacies and other health relevant systems. The urgent efforts are made to integrate all the resources together to provide people the high efficient and networked healthcare information systems. The healthcare information systems also help implement concrete healthcare measurement and treatment, increase service effectiveness and efficiency and reduce cost. In order to make appropriate decision to improve the current healthcare system, healthcare information system is responsible to extract information based on the large amounts of either survey or obtained data. Thus data can be found everywhere but yet been used to find useful information, to discover new knowledge, to develop innovative treatment methods, and to provide intelligent decision making mechanism.

In this study, we first study the demands on healthcare big data. We discuss such big data's characteristics representations in healthcare domain. We presents a the basic Hadoop-based approach on a clinic data to demonstrate Map and Reduce mechanism for distributed file storage utilized in healthcare and clinic services.

II. DEMANDS OF HEALTHCARE SERVICES ON BIG DATA PROCESSING

Big data does not just refer to big volume of data. Its entity conceptually extends to become an advanced technology which includes any methodology and system solutions to dealing with large volume datasets. It has been used as modern computational technique to process, analyze, and mine data which has four special characteristics fast growth rate of generation, variety of structures, and implicit values for information acquisition and knowledge discovery.

The report from McKinsey Global Institute (MGI) in

978-1-5090-0454-6/15 $31.00 © 2015 IEEE
DOI 10.1109/ICICSE.2015.41

2011 clearly emphasized the concept of big data and its potentials in all the areas. In indicates the applications of big data can be driving forces to promote the high rates of production generation and consumers regionally and globally. MGI especially pointed the values of big data only in healthcare can raise up to 300 billion dollars and reduce 80% healthcare cost in USA, if big data technology is properly used, because data and information becomes the third resources vs. materials and human labors. In China, healthcare big data analysis is listed as one of five major enterprise business strategies for national and regional developments.

However, even for relatively-developed regions or countries, there lack systematic design and plans, data sharing, regulations/standard implementations and collaborations. It is impossible to dig out information and knowledge from unprecedented data which are either ignored or wasted. Even for well-established healthcare systems in the developed regions, people often focus their efforts on information systems infrastructure and their uses for operational performance and services. They often pay less attention to analysis the acquired data and mine useful information and knowledge discovery. The new information collection and knowledge refreshment are important in today's intelligent and digital world. They are built upon the existing information systems. There need significant efforts to process, analyze and mine the obtained data, which requires not only data resources, but innovative technology, technical work force who are familiar with existing healthcare systems data, capable to utilizing cutting-edge big data technology, and willing to discover new knowledge and create new ideas.

III. HADOOP-BASED COMPUTATIONAL PRINCIPLE FOR BIG DATA

3.1 Introduction to big data

Big data refers to large and complex dataset on which traditional processing methodologies are difficult to process. These data processes usually include acquisition, pre-/post-processing, analysis, querying or searching, sharing, storage, transfer, visualization, and protection etc. The term has been extended to all these handling techniques for huge data sets with the objective to extract new information, and/or discovery new knowledge and/or make decision. The technical challenges are not only due to huge data volume, but complex data structure. In the past, many most of dataset can be handled using relational database management systems and data analysis can be handle by using small scale computers such as desktop statistics and visualization packages. But current big data are come from various resources from internet, wireless devices, digital equipment or modalities etc. In the perspective of process performance, a facility of massive parallel computers and associated high performance computing (HPC) applications are urgently needed[1,2].

Hadoop is one of technical models of data intensive computations. It was initiated by Yahoo's technical team. It has a MapReduce model which was coded on the Nutch Distributed File System (NDFS). Inspirited by the Yahoo team, Goggle deployed this model to successfully process their big data and continued contribute MapReduce modification[3,4]. Lately Apache took over the development and developed an open-source software toolkits in Java for distributed storage and distributed processing of very large data sets on computer clusters built from commodity hardware.

3.2 Update Apache Hadoop modules

Apache Hadoop has two core components. One is Hadoop Distributed File System (HDFS) and another is open source MapReduce.

Hadoop/HDFS is for data storage. It takes over NDFS's responsibility. It enables to store dataset on various computers or computing nodes which are connected through a fast data exchanger across computer cluster (either an integrated HPC system or a networked distributed system with commodity computers). Hadoop/MapReduce is a software potion which implements the original MapReduce algorithm for data processing. With these cores of Hadoop, one can split files into large blocks to be distributed across networked computers or computing nodes in a cluster. In order to process the distributed blocks of dataset, Hadoop transfers its packaged code to all the participated computers or nodes in parallel. Once the nodes accomplish their data process, they send back to a master system or node for data assembling; thus increase overall performance with high computational efficiency and scalability. The basic idea is similar to conventional parallel computer except it relies on an auxiliary parallel file (data) storage system where data are distributed during computation. The data processing on the nodes are in parallel as distributed computing mode with high-speed networking. In this view, traditional HPC is for computation (CPU) oriented, intensive computing through internal (supercomputing) or external high performance networking (cluster or grid computing), while Hadoop-enhanced computing is for large-scale distributed data intensive computing through internal and external networking. The MapReduce plays

It is very similar to Message Passing Interface (MPI) of partial computational task to each node. Instead, it passes a packed code to enable access distributed data for processing.

Today Hadoop/MapReduce has been widely used in processing large volumes of data in big data technology.

It has been accepted into many computing system architectures, constructions and software implementations.

One of the advantages is a Hadoop/MapReduce application environment can be set up using commodity resources, which makes the system with high scalability. In addition, it has high data protection from data loss, since the processed data are distributed using HDFS-powered distributed storage. Therefore Hadoop-based big data process has three major advantages: scalability, reliability, and efficiency.

Besides of two Hadoop core modules, recently Hadoop developed additional supportive modules: Hadoop Common, and Hadoop Yarn. These two components provide additional resources and utilities in order to better support Hadoop/MapReduce operations. Hadoop Common packages several commonly libraries and utilities which may be needed during Hadoop-based computation. Hadoop YARN is a new-developed resource-management platform which helps manage managing computing resources in a cluster environment, such as job scheduling and monitoring.

IV. MAPREDUCE FOR CLINIC DISTRIBUTED DATA PROCESSING

In order to better understand the mechanism and principle of MapReduce algorithm, we applied the Map and Reduce approaches to a clinic data. The data is about monthly-based college student clinic visits, healthcare service fees, and insurance reimbursement. The original dataset is too large to use MS Excel and we only select few records for the warming up the use of MapReduce.

In this sample test, we select monthly-based data including student ID, basic clinic service fee, medicine fee and special medical treatment fee, and amount of allowed reimbursement. In this simple test, we only calculate the total values. The private information has been filleted out and selected data can be found in Table 1.

Table 1 Student Clinic Data

Student NO.	Clinic Fee	Medicine Fee	Treatment Fee	Total Reimbursement
1	6.40	26.30	0.00	32.70
2	6.40	44.60	0.00	51.00
3	12.80	0.00	208.00	220.80
4	19.20	221.30	24.00	264.50
5	6.40	88.90	418.40	513.70
6	38.40	105.90	904.00	1048.30
7	12.80	24.90	32.00	69.70
8	6.40	213.90	84.80	305.10
09	6.40	104.20	88.00	198.60
10	6.40	36.00	56.00	258.40
11	6.40	76.50	12.00	94.90
12	12.80	3.90	96.00	112.70
13	12.80	0.00	288.00	300.80
14	6.40	0.00	802.40	808.80
15	6.40	34.00	8.00	48.40
16	6.40	33.90	16.00	56.30
17	12.80	22.60	0.00	35.40
18	12.80	37.80	400.00	450.60
19	5.60	29.40	1.60	36.60
20	0.00	37.50	16.00	53.50
21	6.40	0.00	64.00	70.40
22	5.60	0.00	548.00	553.60
23	6.40	0.00	200.00	206.40
24	12.80	117.90	24.80	155.50

Before we establish computational model, we need to preprocess the data and define the data type for each data entity, which are listed in the Table 2.

Table 2. Naming and String Name and Type

	String Name	Type
Student	**No.**studentID	String
Clinic Service Fee	clinicSrviceFee	int
Medicine Fee	medicineFee	int
Treatment Fee	treatmentFee	int
Total Reimbursement	reimbursementbFee	int

Based on Hadoop/MapReduce, one should divide the whole data into M sub datasets each of which can employ mapper and reducer functions for the *M* data communication channels. This process is called Split in MapReduce. For each slipped dataset, one can use map() function built in MapReduce executes its own map decomposition. In way the whole sub dataset in slipped task (say ***m*** task) can be further decomposed into *N* data arrays. For example in our test, for the 24 clinic service fees in the sample of data records, we decompose the data into 4 (N=4) sub datasets and construct 4 vector arrays. They are

<$\mathbf{m}_1$, 6.4, 6.4, 12.8, 19.2, 6.4, 38.4>
<$\mathbf{m}_1$, 12.8, 6.4, 6.4, 6.4, 6.4, 12.8>
<$\mathbf{m}_1$, 12.8, 6.4, 6.4, 6.4, 12.8, 12.8>
<$\mathbf{m}_1$, 5.6, 0.0, 6.4, 5.6, 6.4, 12.8>

Mapper and Reducer Function pseudocode (in Java Code) can be written as

Class Mapper:

```
method map (filename, fileContent)
for each line in fileContent do
studentID=getStudentID(line)
clinicFee=getClinicFee(line)
medicineFee=getMedicineFee(line)
treatmentFee=getTreatmentFee(line)
reimbursementFee=getReimbrusementFee(line)
emit
(studentID,clinicFee,medicineFee,treatmentt
Fee,reimbursementFee)
```

Class Reducer

```
method reduce1(studentID,
[clinicFee1,clinicFee2,…])
totalClinicFee=0
for each clinicFee in [clinicFee1,
clinicFee2,…) do
totalClinicFee+=clinicFee
emit(studentID,totalClinicFee)

method reduce2(studentID,[medicineFee1,
medicineFee2,…])
totalMedicineFee=0
for each medicineFee in [medicineFee1,
medicineFee2,…) do
totalMedicineFee+=medicineFee
emit(studentID,totalMedicineFee)

method reduce3(studentID,[treatmentFee1,
treatmentFee2,…])
totalTreatmentTestFee=0
for each treatmentFee in [treatmentFee1,
treatmentFee2,…) do
totalTreatment+=treatmentFee
emit(studentID,totalTreatmentFee)

method reduce4(studentID,
[reimbursementFee1,reimbursementFee2,…])
totalReimbursementFee=0
for each ReimbursementFee in
[ReimbursementFee1,ReimbursementFee2,…) do
totalReimbursementFee+= ReimbursementFee
emit(studentID, totalReimbursementFee)
```

The Mapper's function map () reads each student's record (student ID, clinic service fee, medicine fee, and treatment fee) and forms 4 vectors. *m* stands for vector name, and the subscript i refers to the procedure index. Through the four reducer functions (reduce1, reduce2, reduce3, and reduce 4) to accumulate each field data in each vector. The functions reduce1(), reduce2(), reduce3(), and reduce4() accumulate each vector's clinic, medicine, treatment and reimbursement fees, respectively. For example, after invoking the function reduce1() for clinic serve fees, the accumulation values from four k_1 vectors give 89.6, 51.2, 57.6, and 38.8, respectively. Thus to form new vectors k_2, which are

$<m_2, 89.6, 51.2>$

$<m_2, 57.6, 38.8>$

Invoking the function reduce1() again, one can construct k_3 vector, $<m_3, 140.8, 96.4>$ and call the reduce1() function once more, one can obtain the final total clinic service fees the clinic received. That is 237.2. The MapReduce workflow can be illustrated in Figure 1 adopted from [8], in which the detailed procedures for invoking each map and reduce functions are clearly depicted.

MapReduce framework has many commonly used modules for counting, classification, filtering, sorting, distinct counting, and cross-counting etc. These functional modules are very useful in dealing with healthcare data. Hereby, we give two examples to demonstrate how these functional modules are used. Only simple count modules and classification modules are given for demonstration.

Many healthcare dataset are composed of a large numbers of patient records. Each record contains many countable fields which often require to be manipulated. For example, we are often account for the total number of visits, male or female patients, children, aged patents, local or visiting patients, revisits, special clinic visits, regular annual or biannual checks, scheduled screens, community service checks, transferred patients, emergency service, various insurance services, cancer patients, diabetes patients, various department visit statistics counting, hospital and clinic registrations status (online or called, or walk in) etc. These calculated counting data can provide quite useful information to understand the healthcare services status, assessment, and qualification, as well as future improvement, monitoring current operational status, design strategic plan and making positive decision etc. al. We need to know the total number of out-patient visits often and in-patient bed occupations in order to manage our healthcare professionals and work forces. In order to count for the total number for each field, we can use map() function to obtain each visit value and then use reduce() function to perform desired calculations. Such counting can be accomplished using MapReduce's Counting target, simply coded as follows.

MapReduce Counting

Class Mapper

```
method map(fileName,fileContent)
for eachline in fileContent do
patientID=getPatientID(line)
visitingNumber=getVistingNumber(line)
emit(patientID,visitingNumber)
```

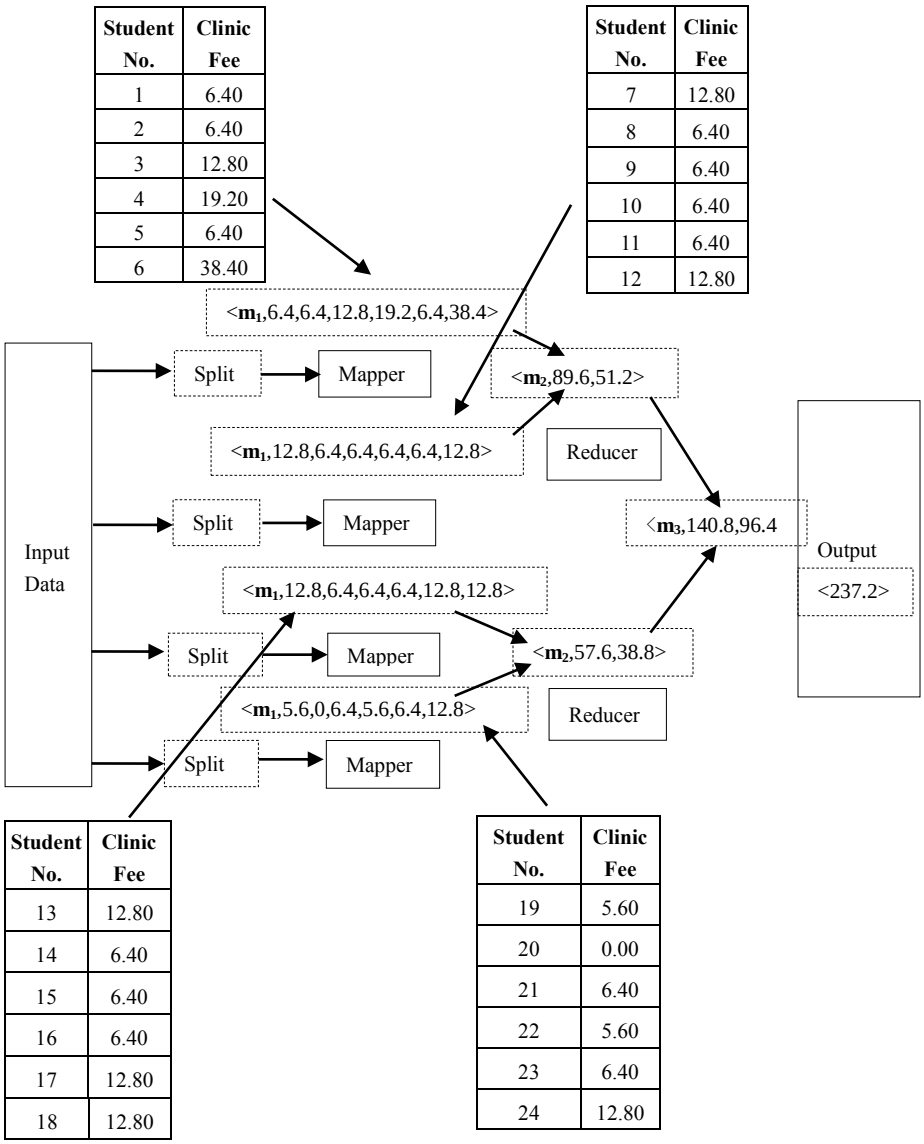

Figure 1 MapReduce mechanism with map() and reduce() functions, the workflow illustration is adopted from [8])

Class Reducer

```
method reduce
(patientID,[visitingNumber1,visitingNumber2
,…])
outputValue=0
for eachpropertyValue in
[visitingNumber1,visitingNumber2,…]do
outputValue=caculateFunction(visitingNumber)
emit(patientID,visitingNumber)
```

In healthcare systems, after obtaining a large number of patient healthcare data, professionals are very interested in knowing some common patient symptoms, possible side effects, various influence factors, therapy results and responses. Most of patients have records large number of records each of which contains various formats of data including inpatient and outpatient data, daily healthcare records, clinic and hospital documentations, image data. These data files are either storage locally in clinic or hospitals at different levels and sometimes geographically located. These data contains a number of properties of each entity. In order to further understand

the distributions and commonality of each healthcare issue or disease, it is necessary to conduct a classification analysis on these big data. For example, from the outpatient in the hospital records, people need know the correlation between weight-height ratio standard and the hypertrophy of patient data stored in a file, so some of properties upon selected entities can be grouped together. Such approach utilizes a classification method. Fortunately, Hadoop/MapReduce provides this kind of module which helps to deal with such data in a parallel and mode. The Mapper's map () function invokes each record in the patient identity records, the patient's height and weight data, and calculate overweight measure value using some standard formula. If the calculated value is above the threshold value, the patient is classified as a special patient for special medical treatment, so healthcare professionals can provide certain preventions for the patient from coronary heart disease or give medical advice to. After grouping or classification, the Hadoop's reduce() function can be invoked to store with the patients data with similar results for subsequent processing. The potential coronary artery disease information can be extracted for dedicated patients.

MapReduce Classification

```
Class Mapper method map
(patientFileName,fileContent)
for eachline infileContent do
item=getItem(line)
Value=givenFunction(item)
emit(overweightValue,line)

classReducer
method reduce(overweightValue,[line1,
line2,…])
file=openFile(overweightValue)
for each line in [line1,line2,…] do
saveToFile(file,line)
```

The other modules can be found in MapReduce package. These computation modules provide useful tools for people to manipulate data in the Hadoop/MapReduce platform. There are many references are available. The only efforts should be made is to deploy rich MapReduce schemes and algorithms to be transformed or translated into the applications for healthcare.

V. CONCLUSION

Healthcare domain has a huge number of patient data. These data have big data's properties. There are urgent demands for healthcare professionals to work on these data which have yet been undiscovered. Rapidly developed big data technology, specially Hadopp/MapReduce big data processing techniques can be used to process and mine these data for new healthcare or medical knowledge discovery, for innovative medical treatment development, and for intelligent decision making. The multiple healthcare data can be streamed in a Hadopp/MapReduce-equipped system to decompose large job into small sets for parallel processing with distributed data storage facility. The split sub dataset can be mapped for calculation and then reduced final output in parallel operations. The MapReduce calculation modules are useful in healthcare big data processing and analysis with counting, classification and mining procedures. Healthcare professionals should work closely with information technology developers and providers to as quick as possible migrate the cutting-edge big data technology to healthcare domain applications.

Reference

[1] A. Jacobs, "The Pathologies of Big Data," *ACMQueue*, July 2009

[2] Roger Magoulas, Ben Lorica, *Introduction to Big Data*, 2009, O'Reilly Media

[3] EMC Education Services, *Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data*, John Wiley & Sons, 2014

[4] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," in *6th Symposium on Operating Systems Design and Operation*, 2004

[5] IBM, "What is the Hadoop Distributed File System (HDFS)?" IBM. 2014. http://www-01.ibm.com/software/data/infosphere/hadoop/hdfs/

[6] Apache Software Foundation, "Resource (Apache Hadoop Main 2.5.1 API)," 2014 https://hadoop.apache.org/

[7] Arun Murthy," Apache Hadoop YARN – Concepts and Applications," 2012

[8] Jun Liu, *Hadoop Big Data Processing*, PTPress, 2013 (in Chinese)