Healthcare Data Analysis using Dynamic Slot Allocation in Hadoop

Aditi Bansal, Ankita Deshpande, Priyanka Ghare, Seema Dhikale, Balaji Bodkhe

Abstract*— *In this new era of big data even health care needs to be modernized, this includes that the health care data should be properly analyzed so that we can deduce that in which group or gender, diseases attack the most. This gigantic size of analytics will need large computation which can be done with help of distributed processing, Hadoop. MapReduce, a popular computing paradigm for large-scale data processing in cloud computing. However, the slot-base MapReduce system (e.g., Hadoop MRv1) due to its unoptimized resource allocation, can suffer from poor performance. To address it, the framework in this paper optimizes the resource allocation. Due to the static pre-configuration of distinct map slots and reduce slots which are not fungible, many a times slots can be severely under-utilized. This is because map slots might be fully utilized while reduce slots may remain empty, and vice-versa. We propose an alternative technique called Dynamic Hadoop Slot Allocation by keeping the slot-based allocation model. It relaxes the slot allocation constraint and allows slots to be reallocated to either map or reduce tasks depending on their needs. The framework's use will provide multipurpose beneficial outputs which include: getting the health care analysis in various forms. Thus this concept of analytics should be implemented with a view of future use.

***Index Terms*—Big data, Hadoop, MapReduce, Slot Allocation**

I. INTRODUCTION

MapReduce is a high performance computing paradigm for large-scale data processing in clusters and data centers, and it has become very popular in recent years. An open source implementation of MapReduce is known as Hadoop. To support batch processing for large jobs submitted from multiple users, companies like Yahoo! and Facebook are deploying Hadoop in large clusters containing thousands of machines. There have been many studies in optimizing MapReduce/Hadoop but, several key challenges for the utilization and performance improvement of a Hadoop cluster are still there.

The paper mainly consists of two phases:

- Big data Analytics in Healthcare
- Dynamic Slot Allocation in a MapReduce cluster.

- ***Big Data Analysis in Healthcare***

For a big data analytics project in healthcare, the conceptual framework is similar to that of a traditional health informatics or analytics project. But, how processing is executed is totally different in both.

The analysis can be performed with a business intelligence tool installed on a stand-alone system, like a laptop or a desktop in a regular health analytics project. Because according to the definition, big data is nothing but large processing distributed across the multiple nodes. This concept has existed for many years. To gain insight for making better-informed health-related decisions, as healthcare providers start to tap into their large data repositories, relatively new thing is its use in analyzing very large data sets. Also, the applications of big data analytics in healthcare have been encouraged by open source platforms such as Hadoop/MapReduce, available on the cloud. The typical end-user in healthcare may not possess the skills required by these platforms or tools such as, a great deal of programming. Governance issues including ownership, standards, privacy and security have yet to be addressed, taking into consideration the only recent emergence of big data analytics in healthcare. To develop and implement a big data project for healthcare providers, an applied big data analytics in healthcare methodology is offered by them in [2].

- ***Scheduling and Resource Allocation Optimization.***

MapReduce jobs include some existing computation scheduling and resource allocation optimization work. In many real-world applications, the execution time for map and reduce tasks for each job may not be available, though it is assumed that it should be known in advance. It fails to consider the jobs with dependency, and is only suitable for independent jobs. A resource stealing method to enable running tasks to steal resources reserved for idle slots and give them back proportionally whenever new tasks are assigned, by adopting multithreading technique for running tasks on multiple CPU cores has been proposed by Guo et al. [5]. For the utilization improvement of the purely idle slave nodes without any running tasks, it cannot work.

II. BIG DATA OVERVIEW

We are awash in a flood of data today. The data is being generated so rapidly that now it is a need to manage this data in an intelligent way. Previously, decisions were based on guess work but today with great development in the field of analytics the decisions can be taken based on data itself.

Manuscript Received on November 2014.

Aditi Bansal, Department of Computer, Modern Education Society's College of Engineering ,University of Pune, Pune, India.
Ankita Deshpande, Department of Computer, Modern Education Society's College of Engineering ,University of Pune, Pune, India.
Priyanka Ghare, Department of Computer, Modern Education Society's College of Engineering ,University of Pune, Pune, India.
Seema Dhikale, Department of Computer, Modern Education Society's College of Engineering ,University of Pune, Pune, India.
Prof. Balaji Bodkhe, Department of Computer, Modern Education Society's College of Engineering ,University of Pune, Pune, India.

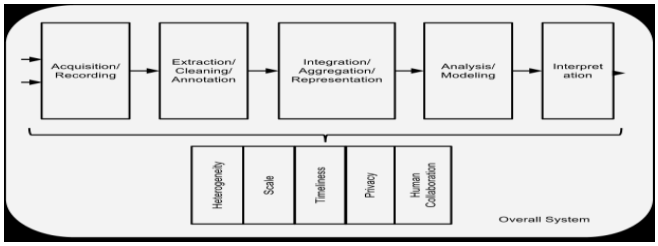

Fig. 1: Phases in Big Data

The figure above describes the phases in big data[3]. We have five phases which are described as follows:

(i) Data Acquisition and Recording

Big data is not generated out of vacuum. It is generated from data generating source.

These sources can produce terabytes of raw data per day. This data need to be filtered in such a way that no useful information should be discarded.

(ii) Information Extraction and Cleaning

The recorded information is not in the format ready for analysis. We require an information extraction process that retrieves the required information from the recorded data and express that information in a format which is suitable for analysis.

(iii) Timeliness

Speed is directly proportional to time. The larger the data set the longer it will take to analyse. The design of the system should be such that it deals with size effectively and can process the data with a faster speed.

(iv) Privacy

The privacy of data is major concern. The security of this massive amount of data is very necessary since if this data goes into wrong hands blunder can occur.

(v) Human Collaboration

With the tremendous advances in the computational analysis, there still many patterns that humans can easily detect but computer algorithms fail to do so. Analytics for big data will not all computational, rather it will have a human in the loop. Big data analysis system must support the input from human experts. These data system should accept this distributed input and should support collaboration. There are challenges still present in big data but in the near future these challenges will be solved and we will be able to achieve the promised benefits of big data.

III. BASICS OF MAPREDUCE

Some basic ideas of MapReduce and relative features of Hadoop are reviewed in this section. Google first introduced a popular parallel model known as MapReduce. MapReduce is designed to handle and generate large scale data sets in distributed environment. A convenient way to parallelize data analysis process is also provided by MapReduce. It has some advantages such as conveniences, robustness, and scalability. All the map tasks (and reduce tasks) are executed in a fully parallel way in a MapReduce system. Therefore, through the use of the MapReduce model, high- level parallelism can be achieved for data processing. The basic idea of MapReduce is to split the large input data set into many small pieces and assign small tasks to multiple devices in a distributed environment. The process of MapReduce includes two major parts, Map function and Reduce function. First of all, the input files will be automatically split and copied to different computing nodes. Later on, the inputs in key-value pair format will be sent to Map function. The input pairs will be processed by map function and it will also generate intermediate key-value pairs which will be inputs for Reduce function. The inputs which have the same key are combined and the final result will be generated by reduce function. The final result will be written into the distributed file system. Only the Map and Reduce functions are needed to be implemented by users. The partitioning of the inputs is automatically done by the model, users do not have to worry about it. The tasks will be assigned evenly to every machine by the model. Hadoop is an open source software framework which implements MapReduce model. The components of the MapReduce framework are: 1) Job Tracker, 2) Task Tracker, 3) Name Node and 4) Data Node.

The Name Node stores the file system metadata. This metadata includes which files are maps to which block locations and which blocks are stored on which data node. The actual data resides in the data node. The heartbeat messages to name node every 3 seconds are sent by all data nodes to say that data nodes are alive. If heartbeat message is not received by name node from a data node for 10 minutes, then it assumes that data node is dead. To rebalance the data, move and copy, all the data nodes communicate with each other. The Job Tracker is used to managing the Task tracker and resource management that is tracking resource availability and time management of each job. A number of tasks is pre-configured in Task Tracker. The Job Tracker consists of Job History. Fig 2 represents the framework of MapReduce[4]. It consists of different components namely Name Node, Job Tracker and Task Tracker. The file in a distribute file system is stored in the Name Node. The resource availability and resource management of MapReduce framework is monitored by the Job Tracker. The Job Tracker consists of two phases. 1) Logs and Job History. Job History contains the past job description. It also provides different parameters like number of nodes in a cluster, number of the jobs, job Id, execution time and memory usage of each job etc.

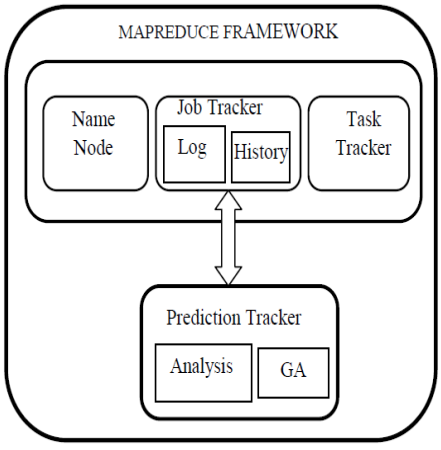

Fig. 2: Mapreduce Framework

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

IV. BIG DATA ANALYSIS IN HEALTHCARE

The first phase of the paper consists of the healthcare analysis phase. The health care industry is generating large amounts of data. While most of the data is in hard copy format, the latest trend is to move towards digitization of this massive volume of data. Reports say that in U.S. alone the healthcare data reached upto 150 exabytes in 2011. Big data in healthcare is overwhelming not only because it has massive volume but

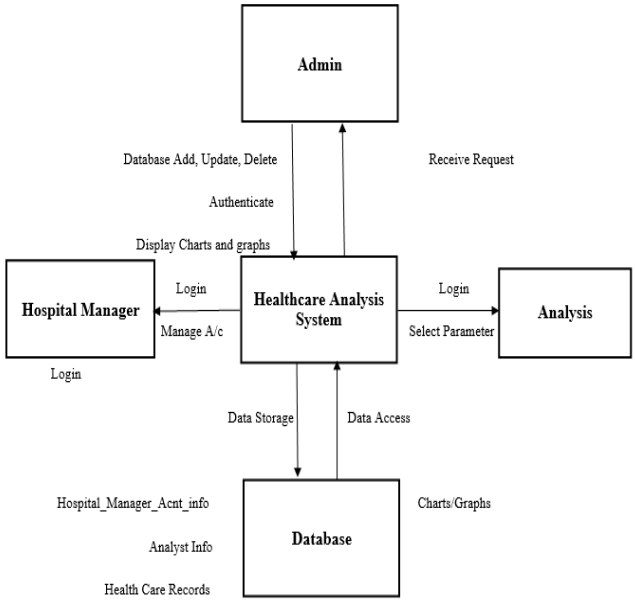

Fig. 3: System Architecture

also because of the diversity of data types and the speed at which it must be managed[2]. In this paper we will see that how an analyst or researcher can play with this voluminous data and get the desired analysis in chart or graphical format. The system we talking of will consist of various users namely hospital manager, admin, analyst/researcher. The hospital manager will be responsible to add the data in a particular format. He logs in with his unique hospital id which after getting authenticated by the admin gets access for entering the medical records. The role of the analyst/researcher is to select the parameters for the analysis. The parameters can be in the form of dates, gender or year. Once he selects the parameter, he can select the display method and after the all the selection he can get the desired output. We are sure that this type of health care analysis will surely help the hospital manager to keep track of their records as well as the analyst will be able to do the analysis in a more organised way. Big data analytics and applications in healthcare are at a nascent stage of development, but advances in platforms and tools can accelerate their maturing process to a greater extent.

V. DYNAMIC HADOOP SLOT ALLOCATION (DHSA)

DHSA keeps the slot-based resource model in contrast to YARN. Both map and reduce tasks can run on new resource model of 'container', proposed by YARN. The idea for DHSA is to break the assumption of slot allocation constraint to allow that:

1. Either map or reduce tasks can use the slots and slots are generic, even if number of map and reduce slots are pre-configured. In other words, if number of map tasks is greater than map slots, then map tasks can borrow the unused reduce slots and vice-versa.

2. Although map tasks can use unused reduce slots, map tasks will prefer to use map slots. Similarly, even though reduce tasks can use unallocated map slots in case of insufficient reduce slots, reduce tasks prefer to use reduce slots. But, to control the ratio of running map and reduce tasks during runtime, the pre-configuration of map and reduce slots per slave node can still work. This is the main benefit and it is better than YARN. YARN has no control mechanism for the ratio of running map and reduce tasks. Because, too many reduce tasks running for data shuffling, cause the network to be a bottleneck without control.

With DHSA, both map and reduce tasks can be run on either map or reduce slots. However challenges like fairness should be considered. Fairness is an important metric in Hadoop Fair Scheduler (HFS). We say it is fair when all pools have been allocated with the same amount of resources. In HFS, task slots are first allocated across the pools, and then the slots are allocated to the jobs within the pool. DHSA contains two alternatives as: 1) Pool-Independent DHSA (PI-DHSA) and 2) Pool-dependent DHSA (PD-DHSA). Both of these consider the fairness from different aspects.

1. Pool-Independent DHSA (PI-DHSA)

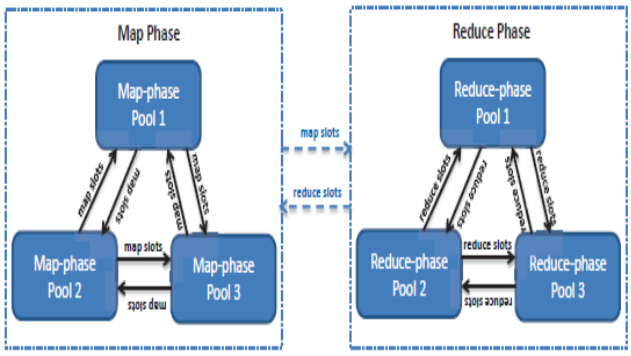

Fig. 4: Pool Independent DHSA

Fig 4 represents the slot allocation flow for PI-DHSA. To allocate slots across pools with minimum guarantees at the map-phase and reduce phase, respectively, HFS adopts Max-min fairness. Pool-Independent DHSA (PI-DHSA) extends the HFS by allocating slots from the cluster global level, independent of pools. When the numbers of typed slots allocated across typed-pools within each phase are the same, it considers fair.

2. Pool-Dependent DHSA (PD-DHSA)

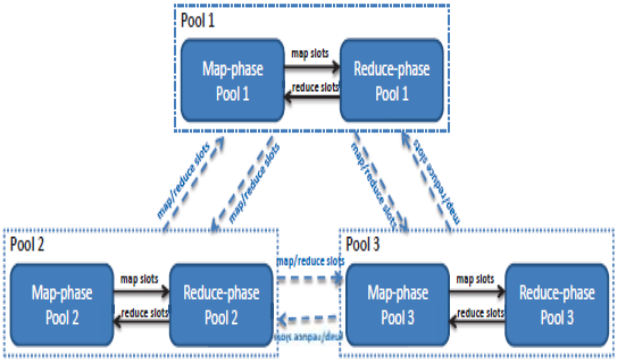

Fig. 5: Pool Dependent DHSA

Fig 5 represents an example of the fairness-based slot allocation flow for PD-DHSA. In the diagram, the black arrow line and dash line show the borrow flow for slots across pools. Pool-Dependent DHSA (PD-DHSA) considers fairness for the dynamic slot allocation across pools, as shown in figure, in contrast to PI-DHSA that considers the fairness in its dynamic slot allocation independent of pools. When total numbers of map and reduce slots allocated across pools are the same with each other, it considers fair.

A. Abbreviations and Acronyms

DHSA: Dynamic Hadoop Slot Allocation
PI-DHSA: Pool Independent DHSA
PD-DHSA: Pool Dependent DHSA

VI. CONCLUSION

The potential of big data is to transform the way healthcare providers use sophisticated technologies to gain knowledge from their clinical and other data sources and make good decisions [2]. In the near future we will see rapid and widespread implementation of big data analytics in health care industry. Big data analytics guarantees privacy and security. The applications of big data analytics are still at nascent stage of development and its implementation in the health care industry will surely help its organizations. This paper proposes a framework which is aiming that it will improve the performance of MapReduce workloads and at the same time will maintain the fairness. DHSA [1] the technology about which we have mentioned above focuses on the maximum utilization of slots by allocating map (or reduce) slots to map and reduce tasks dynamically. There are two types of DHSA namely, PI –DHSA and the other is PD-DHSA they differ in the levels of fairness and user can choose any among them according to their requirements.

ACKNOWLEDGMENT

We would like to take this opportunity to express our deepest gratitude to all those who have supported us and helped us to make this paper a reality. We are highly indebted to **Prof. Balaji Bodkhe** for his guidance and constant supervision as well as for providing necessary information regarding the paper and also for his support in completing the paper. Furthermore, we would like to acknowledge the crucial role of the University of Pune and our principal **Prof. A. A. Keste**, for providing us with the privilege to undergo this enriching experience. In conclusion, we would like to thank our parents and colleagues for their kind co-operation and investing their time, and willingly helping us out with their abilities to make this paper a reality.

REFERENCES

1. Shanjiang Tang, Bu-Sung Lee, Bingsheng He, “DynamicMR: A Dynamic Slot Allocation Optimization Framework for MapReduce Clusters”, *IEEE Transactions, 2013.*
2. Wullianallur Raghupathi and Viju Raghupathi, “Big data analytics in healthcare: promise and potential”, *Health Information Science and Systems 2014.*
3. Divyakant Agrawal, UC Santa Barbara, Philip Bernstein, Microsoft Elisa Bertino, Purdue Univ. ” Big Data White pdf”, from Nov 2011 to Feb 2012
4. V.Sivaranjani, R.Jayamala, “Optimization of Workload Prediction Based on MapReduce Frammework in a Cloud System”, *IJRET, 2014.*
5. Z.H. Guo, G. Fox, M. Zhou, Y. Ruan., “Improving Resource Utilization in MapReduce”, *IEEE Cluster’12.*
6. Tom White, “Hadoop: The Definitive Guide”.
7. Chuck Lam, “Hadoop in Action”.