A Review Paper on Big Data and Hadoop

Harshawardhan S. Bhosale[1], Prof. Devendra P. Gadekar[2]

[1]Department of Computer Engineering,
JSPM's Imperial College of Engineering & Research, Wagholi, Pune
Bhosale.harshawardhan186@gmail.com

[2] Department of Computer Engineering,
JSPM's Imperial College of Engineering & Research, Wagholi, Pune
devendraagadekar84@gmail.com

Abstract: The term 'Big Data' describes innovative techniques and technologies to capture, store, distribute, manage and analyze petabyte- or larger-sized datasets with high-velocity and different structures. Big data can be structured, unstructured or semi-structured, resulting in incapability of conventional data management methods. Data is generated from various different sources and can arrive in the system at various rates. In order to process these large amounts of data in an inexpensive and efficient way, parallelism is used. Big Data is a data whose scale, diversity, and complexity require new architecture, techniques, algorithms, and analytics to manage it and extract value and hidden knowledge from it. Hadoop is the core platform for structuring Big Data, and solves the problem of making it useful for analytics purposes. Hadoop is an open source software project that enables the distributed processing of large data sets across clusters of commodity servers. It is designed to scale up from a single server to thousands of machines, with a very high degree of fault tolerance.

Keywords -*Big Data, Hadoop, Map Reduce, HDFS, Hadoop Components*

1. Introduction

A. Big Data: Definition

Big data is a term that refers to data sets or combinations of data sets whose size (volume), complexity (variability), and rate of growth (velocity) make them difficult to be captured, managed, processed or analyzed by conventional technologies and tools, such as relational databases and desktop statistics or visualization packages, within the time necessary to make them useful. While the size used to determine whether a particular data set is considered big data is not firmly defined and continues to change over time, most analysts and practitioners currently refer to data sets from 30-50 terabytes(10 12 or 1000 gigabytes per terabyte) to multiple petabytes (1015 or 1000 terabytes per petabyte) as big data. Figure No. 1.1 gives Layered Architecture of Big Data System. It can be decomposed into three layers, including Infrastructure Layer, Computing Layer, and Application Layer from top to bottom.

B. 3 Vs of Big Data

Volume of data: Volume refers to amount of data. Volume of data stored in enterprise repositories have grown from megabytes and gigabytes to petabytes.

Variety of data: Different types of data and sources of data. Data variety exploded from structured and legacy data stored in enterprise repositories to unstructured, semi structured, audio, video, XML etc.

Velocity of data: Velocity refers to the speed of data processing. For time-sensitive processes such as catching fraud, big data must be used as it streams into your enterprise in order to maximize its value.

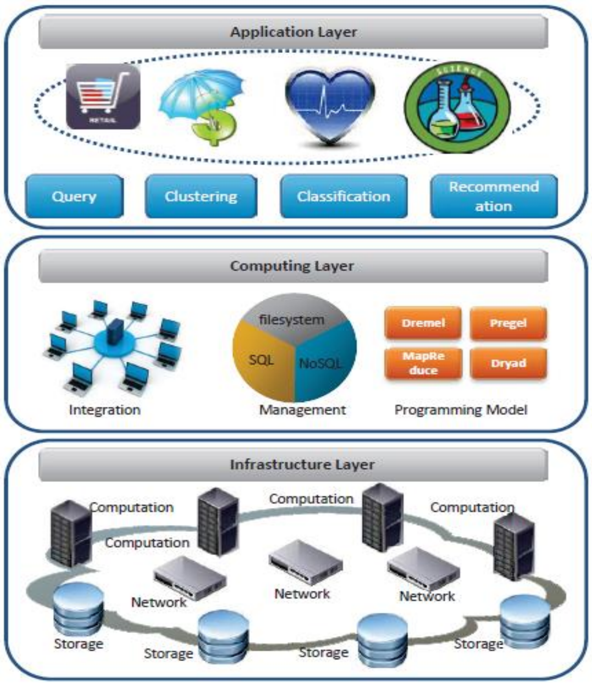

Figure 1: Layered Architecture of Big Data System

C. Problem with Big Data Processing

i. Heterogeneity and Incompleteness

When humans consume information, a great deal of heterogeneity is comfortably tolerated. In fact, the nuance and richness of natural language can provide valuable depth. However, machine analysis algorithms expect homogeneous data, and cannot understand nuance. In consequence, data must be carefully structured as a first step in (or prior to) data analysis. Computer systems work most efficiently if they can store multiple items that are all identical in size and structure. Efficient representation, access, and analysis of semi-structured data require further work.

ii. Scale

Of course, the first thing anyone thinks of with Big Data is its size. After all, the word "big" is there in the very name. Managing large and rapidly increasing volumes of data has been a challenging issue for many decades. In the past, this challenge was mitigated by processors getting faster, following Moore's law, to provide us with the resources needed to cope with increasing volumes of data. But, there is a fundamental shift underway now: data volume is scaling faster than compute resources, and CPU speeds are static.

iii. Timeliness

The flip side of size is speed. The larger the data set to be processed, the longer it will take to analyze. The design of a system that effectively deals with size is likely also to result in a system that can process a given size of data set faster. However, it is not just this speed that is usually meant when one speaks of Velocity in the context of Big Data. Rather, there is an acquisition rate challenge

iv. Privacy

The privacy of data is another huge concern, and one that increases in the context of Big Data. For electronic health records, there are strict laws governing what can and cannot be done. For other data, regulations, particularly in the US, are less forceful. However, there is great public fear regarding the inappropriate use of personal data, particularly through linking of data from multiple sources. Managing privacy is effectively both a technical and a sociological problem, which must be addressed jointly from both perspectives to realize the promise of big data.

v. Human Collaboration

In spite of the tremendous advances made in computational analysis, there remain many patterns that humans can easily detect but computer algorithms have a hard time finding. Ideally, analytics for Big Data will not be all computational rather it will be designed explicitly to have a human in the loop. The new sub-field of visual analytics is attempting to do this, at least with respect to the modeling and analysis phase in the pipeline. In today's complex world, it often takes multiple experts from different domains to really understand what is going on. A Big Data analysis system must support input from multiple human experts, and shared exploration of results. These multiple experts may be separated in space and time when it is too expensive to assemble an entire team together in one room. The data system has to accept this distributed expert input, and support their collaboration.

2. Hadoop: Solution for Big Data Processing

Hadoop is a Programming framework used to support the processing of large data sets in a distributed computing environment. Hadoop was developed by Google's MapReduce that is a software framework where an application break down into various parts. The Current Appache Hadoop ecosystem consists of the Hadoop Kernel, MapReduce, HDFS and numbers of various components like Apache Hive, Base and Zookeeper. HDFS and MapReduce are explained in following points.

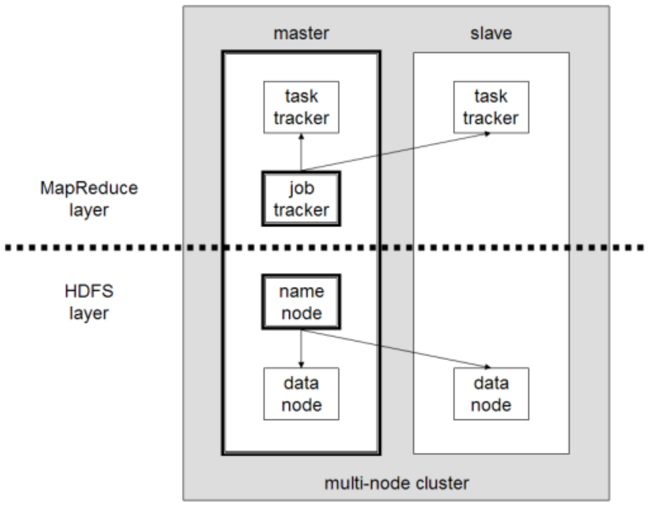

Figure 2: Hadoop Architecture

A. HDFS Architecture

Hadoop includes a fault-tolerant storage system called the Hadoop Distributed File System, or HDFS. HDFS is able to store huge amounts of information, scale up incrementally and survive the failure of significant parts of the storage infrastructure without losing data. Hadoop creates *clusters* of machines and coordinates work among them. Clusters can be built with inexpensive computers. If one fails, Hadoop continues to operate the cluster without losing data or interrupting work, by shifting work to the remaining machines in the cluster. HDFS manages storage on the cluster by breaking incoming files into pieces, called "blocks," and storing each of the blocks redundantly across the pool of servers. In the common case, HDFS stores three complete copies of each file by copying each piece to three different servers.

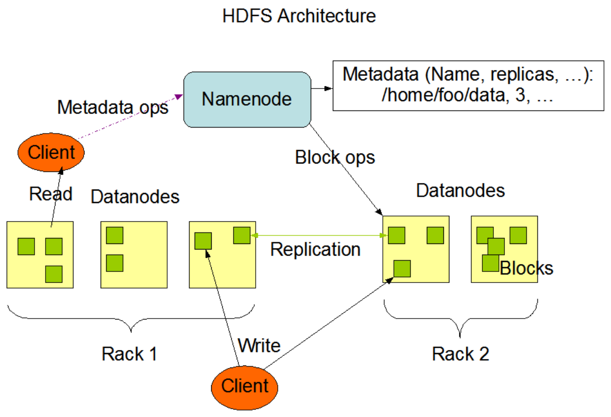

Figure 3: HDFS Architecture

B. MapReduce Architecture

The processing pillar in the Hadoop ecosystem is the MapReduce framework. The framework allows the specification of an operation to be applied to a huge data set, divide the problem and data, and run it in parallel. From an analyst's point of view, this can occur on multiple dimensions. For example, a very large dataset can be reduced into a smaller subset where analytics can be applied. In a traditional data warehousing scenario, this

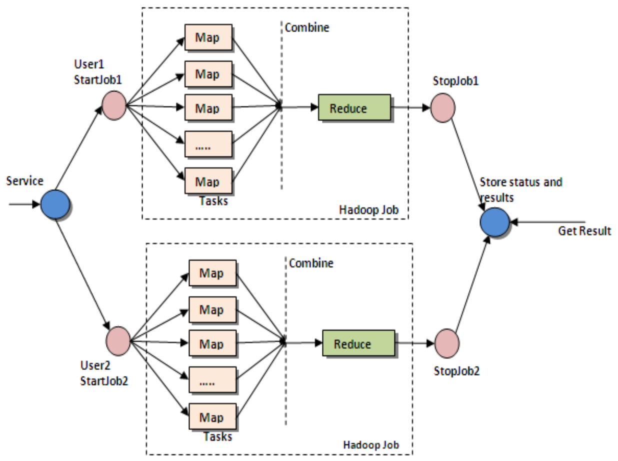

Figure 4: MapReduce Architecture

might entail applying an ETL operation on the data to produce something usable by the analyst. In Hadoop, these kinds of operations are written as MapReduce jobs in Java. There are a number of higher level languages like Hive and Pig that make writing these programs easier. The outputs of these jobs can be written back to either HDFS or placed in a traditional data warehouse. There are two functions in MapReduce as follows:

map – the function takes key/value pairs as input and generates an intermediate set of key/value pairs

reduce – the function which merges all the intermediate values associated with the same intermediate key

3. Literature Review

S. Vikram Phaneendra & E. Madhusudhan Reddy et.al. Illustrated that in olden days the data was less and easily handled by RDBMS but recently it is difficult to handle huge data through RDBMS tools, which is preferred as "big data". In this they told that big data differs from other data in 5 dimensions such as volume, velocity, variety, value and complexity. They illustrated the hadoop architecture consisting of name node, data node, edge node, HDFS to handle big data systems. Hadoop architecture handle large data sets, scalable algorithm does log management application of big data can be found out in financial, retail industry, health-care, mobility, insurance. The authors also focused on the challenges that need to be faced by enterprises when handling big data: - data privacy, search analysis, etc [1].

Kiran kumara Reddi & Dnvsl Indira et.al. Enhanced us with the knowledge that Big Data is combination of structured , semi-structured ,unstructured homogenous and heterogeneous data .The author suggested to use nice model to handle transfer of huge amount of data over the network .Under this model, these transfers are relegated to low demand periods where there is ample ,idle bandwidth available . This bandwidth can then be repurposed for big data transmission without impacting other users in system. The Nice model uses a store –and-forward approach by utilizing staging servers. The model is able to accommodate differences in time zones and variations in bandwidth. They suggested that new algorithms are required to transfer big data and to solve issues like security, compression, routing algorithms [2].

Jimmy Lin et.al. used Hadoop which is currently the large –scale data analysis " hammer" of choice, but there exists classes of algorithms that aren't " nails" in the sense that they are not particularly amenable to the MapReduce programming model . He focuses on the simple solution to find alternative non-iterative algorithms that solves the same problem. The standard MapReduce is well known and described in many places .Each iteration of the pagerank corresponds to the MapReduce job. The author suggested iterative graph, gradient descent & EM iteration which is typically implemented as Hadoop job with driven set up iteration &Check for convergences. The author suggests that if all you have is a hammer, throw away everything that's not a nail [3].

Wei Fan & Albert Bifet et.al. Introduced Big Data Mining as the capability of extracting Useful information from these large datasets or streams of data that due to its Volume, variability and velocity it was not possible before to do it. The author also started that there are certain controversy about Big Data. There certain tools for processes. Big Data as such hadoop, strom, apache S4. Specific tools for big graph mining were PEGASUS & Graph. There are certain Challenges that need to death with as such compression, visualization etc.[4].

Albert Bifet et.al. Stated that streaming data analysis in real time is becoming the fastest and most efficient way to obtain useful knowledge, allowing organizations to react quickly when problem appear or detect to improve performance. Huge amount of data is created everyday termed as " big data". The tools used for mining big data are apache hadoop, apache big, cascading, scribe, storm, apache hbase, apache mahout, MOA, R, etc. Thus, he instructed that our ability to handle many exabytes of data mainly dependent on existence of rich variety dataset, technique, software framework [5].

Bernice Purcell et.al. Started that Big Data is comprised of large data sets that can't be handle by traditional systems. Big data includes structured data, semi-structured and unstructured data. The data storage technique used for big data includes multiple clustered network attached storage (NAS) and object based storage. The Hadoop architecture is used to process unstructured and semi-structured using map reduce to locate all relevant data then select only the data directly answering the query. The advent of Big Data has posed opportunities as well challenges to business [6].

Sameer Agarwal et.al. Presents a BlinkDB, a approximate query engine for running interactive SQL queries on large volume of data which is massively parallel. BlinkDB uses two key ideas: (1) an adaptive optimization framework that builds and maintains a set of multi-dimensional stratified samples from original data over time, and (2) A dynamic sample selection strategy that selects an appropriately sized sample based on a query's accuracy or response time requirements [7].

Yingyi Bu et.al. Used a new technique called as HaLoop which is modified version of Hadoop MapReduce Framework, as Map Reduce lacks built-in-support for iterative programs HaLoop allows iterative applications to be assembled from existing Hadoop programs without modification, and significantly improves their efficiency by providing interiteration caching mechanisms and a loop-aware scheduler to exploit these caches. He presents the design, implementation, and evaluation of HaLoop, a novel parallel and distributed system that supports large-scale iterative data analysis applications. HaLoop is built on top of Hadoop and extends it with a new programming model and several important optimizations that include (1) a loop-aware task scheduler, (2) loop-invariant data caching, and (3) caching for efficient fix point verification [8].

Shadi Ibrahim et.al. Project says presence of partitioning skew1 causes a huge amount of data transfer during the shuffle phase and leads to significant unfairness on the reduce input among different data nodes In this paper, author develop a novel algorithm named LEEN for locality aware and fairness-aware key partitioning in MapReduce. LEEN embraces an asynchronous map and reduce scheme. Author has integrated LEEN into Hadoop. His experiments demonstrate that LEEN can efficiently achieve higher locality and reduce the amount of shuffled data. More importantly, LEEN guarantees fair distribution of the reduce inputs. As a result, LEEN achieves a performance improvement of up to 45% on different workloads. To tackle all this he presents a present a technique for Handling Partitioning Skew in MapReduce using LEEN [9].

Kenn Slagter et.al. Proposes an improved partitioning algorithm that improves load balancing and memory consumption. This is done via an improved sampling algorithm and partitioner. To evaluate the proposed algorithm, its performance was compared against a state of the art partitioning mechanism employed by Tera Sort as the performance of MapReduce strongly depends on how evenly it distributes this workload. This can be a challenge, especially in the advent of data skew. In MapReduce, workload distribution depends on the algorithm that partitions the data. One way to avoid problems inherent from data skew is to use data sampling. How evenly the partitioner distributes the data depends on how large and representative the sample is and on how well the samples are analyzed by the partitioning mechanism. He uses an improved partitioning mechanism for optimizing massive data analysis using MapReduce for evenly distribution of workload [10].

Ahmed Eldawy et.al. presents the first full-fledged MapReduce framework with native support for spatial data that is spatial data Spatial Hadoop pushes its spatial constructs in all layers of Hadoop, namely, language, storage, MapReduce and operations layers. In the language layer, a simple high level language is provided to simplify spatial data analysis for non-technical users. In the storage layer, a two-layered spatial index structure is provided where the *global* index partitions data across nodes while the *local* index organizes data in each node. This structure is used to build a grid index, an R-tree or an R+-tree. Spatial-Hadoop is a comprehensive extension to Hadoop that pushes spatial data inside the core functionality of Hadoop. Spatial Hadoop runs existing Hadoop programs as is, yet, it achieves order(s) of magnitude better performance than Hadoop when dealing with spatial data. SpatialHadoop employs a simple spatial high level language, a two-level spatial index structure, basic spatial components built inside the MapReduce layer, and three basic spatial operations: range queries, k-NN queries, and spatial join. Author presents an efficient MapReduce framework for Spatial Data [11].

Jeffrey Dean et.al. Implementation of MapReduce runs on a large cluster of commodity machines and is highly scalable: a typical MapReduce computation processes many terabytes of data on thousands of machines. Programmers and the system easy to use: hundreds of MapReduce programs have been implemented and upwards of one thousand MapReduce jobs are executed on Google's clusters every day. Programs written in this functional style are automatically parallelized and executed on a large cluster of commodity machines. The run-time system takes care of the details of partitioning the input data, scheduling the program's execution across a set of machines, handling machine failures, and managing the required inter-machine Communication. This allows programmers without any experience with parallel and distributed systems to easily utilize the resources of a large distributed system. Author proposes Simplified Data Processing on Large Clusters [12].

Chris Jermaine et.al. Proposes a Online Aggregation for Large-Scale Computing. Given the potential for OLA to be newly relevant, and given the current interest on very large-scale, data-oriented computing, in this paper we consider the problem of providing OLA in a shared-nothing environment. While we concentrate on implementing OLA on top of a MapReduce engine, many of author’s most basic project contributions are not specific to MapReduce, and should apply broadly. Consider how online aggregation can be built into a MapReduce system for large-scale data processing. Given the MapReduce paradigm’s close relationship with cloud computing (in that one might expect a large fraction of MapReduce jobs to be run in the cloud), online aggregation is a very attractive technology. Since large-scale cloud computations are typically pay-as-you-go, a user can monitor the accuracy obtained in an online fashion, and then save money by killing the computation early once sufficient accuracy has been obtained [13].

Tyson Condie et.al. propose a modified MapReduce architecture in which intermediate data is pipelined between operators, while preserving the programming interfaces and fault tolerance models of other MapReduce frameworks. To validate this design, author developed the Hadoop Online Prototype (HOP), a pipelining version of Hadoop. Pipelining provides several important advantages to a MapReduce framework, but also raises new design challenges. To simplify fault tolerance, the output of each MapReduce task and job is materialized to disk before it is consumed. In this demonstration, we describe a modified MapReduce architecture that allows data to be pipelined between operators. This extends the MapReduce programming model beyond batch processing, and can reduce completion times and improve system utilization for batch jobs as well. We demonstrate a modified version of the Hadoop MapReduce framework that supports online aggregation, which allows users to see “early returns” from a job as it is being computed. Our Hadoop Online Prototype (HOP) also supports continuous queries, which enable MapReduce programs to be written for applications such as event monitoring and stream processing [14].

Jonathan Paul Olmsted et.al. Derive the necessary results to apply variation Bayesian inference to the ideal point model. This deterministic, approximate solution is shown to produce comparable results to those from standard estimation strategies. However, unlike these other estimation approaches, solving for the (approximate) posterior distribution is rapid and easily scales to 'big data'. Inferences from the variation Bayesian approach to ideal point estimation are shown to be equivalent to standard approaches on modestly-sized roll call matrices from recent sessions of the US Congress. Then, the ability of variation inference to scale to big data is demonstrated and contrasted with the performance of standard approaches.[15]

Jonathan Stuart Ward et.al. did a survey of Big data definition, Anecdotally big data is predominantly associated with two ideas: data storage and data analysis. Despite the sudden Interest in big data, these concepts are far from new and have long lineages. This, therefore, raises the question as to how big data is notably different from conventional data processing techniques. For rudimentary insight as to the answer to this question one need look no further than the term big data. \Big" implies significance, complexity and challenge. Unfortunately the term\big" also invites quantification and therein lies the difficulty in furnishing a definition. The lack of a consistent definition introduces ambiguity and hampers discourse relating to big data. This short paper attempts to collate the various definitions which have gained some degree of traction and to furnish a clear and concise definition of an otherwise ambiguous term [16].

Albert Bifet et.al. Discuss the current and future trends of mining evolving data streams, and the challenges that the field will have to overcome during the next years. Data stream real time analytics are needed to manage the data currently generated, at an ever increasing rate, from such applications as: sensor networks, measurements in network monitoring and traffic management, log records or click-streams in web exploring, manufacturing processes, call detail records, email, blogging, twitter posts and others. In fact, all data generated can be considered as streaming data or as a snapshot of streaming data, since it is obtained from an interval of time. Streaming data analysis in real time is becoming the fastest and most efficient way to obtain useful knowledge from what is happening now, allowing organizations to react quickly when problems appear or to detect new trends helping to improve their performance. Evolving data streams are contributing to the growth of data created over the last few years. We are creating the same quantity of data every two days, as we created from the dawn of time up until 2003. Evolving data streams methods are becoming a low-cost, green methodology for real time online prediction and analysis [17].

Mrigank Mridul, Akashdeep Khajuria, Snehasish Dutta, Kumar N. et.al did the analysis of big data he stated that Data is generated through many sources like business processes, transactions, social networking sites, web servers, etc. and remains in structured as well as unstructured form . Today's business applications are having enterprise features like large scale, data-intensive, web-oriented and accessed from diverse devices including mobile devices. Processing or analyzing the huge amount of data or extracting meaningful information is a challenging task. The term "Big data" is used for large data sets whose size is beyond the ability of commonly used software tools to capture, manage, and process the data within a tolerable elapsed time. Big data sizes are a constantly moving target currently ranging from a few dozen terabytes to many peta bytes of data in a single data set. Difficulties include capture, storage, search, sharing, analytics and visualizing. Typical examples of big data found in current scenario includes web logs, RFID generated data, sensor networks, satellite and geo-spatial data, social data from social networks, Internet text and documents, Internet search indexing, call detail records, astronomy, atmospheric science, genomics, biogeochemical, biological, and other complex and/or interdisciplinary scientific project, military. Surveillance, medical records, photography archives, video archives, and large-scale ecommerce [18].

Kyong-Ha Lee Hyunsik Choi et.al. Proposes a prominent parallel data processing tool MapReduce survey intends to assist the database and open source communities in understanding various technical aspects of the MapReduce framework. In this survey, we characterize the MapReduce framework and discuss its inherent pros and cons. We then introduce its optimization strategies reported in the recent literature. author also discuss the open issues and challenges raised on parallel data analysis with MapReduce [19].

Chen He Ying Lu David Swanson et.al develops a new MapReduce scheduling technique to enhance map task's data locality. He has integrated this technique into Hadoop default FIFO scheduler and Hadoop fair scheduler. To evaluate his technique, he compares not only MapReduce scheduling algorithms with and without his technique but also with an existing data locality enhancement technique (i.e., the delay algorithm developed by Facebook). Experimental results show that his technique often leads to the highest data locality rate and the lowest response time for map tasks. Furthermore, unlike the delay algorithm, it does not require an intricate parameter tuning process [20].

4. Other Components of Hadoop

The Table 1, Comparison among Components of Hadoop, gives details of different Hadoop Components which have been used now days. HBase, Hive, MongoDB, Redis, Cassandra and Drizzle are the different components. Comparison among these components is done on the basis of Concurrency, Durability, Replication Method, Database Model and Consistency Concepts used in the components.

Table 1: Comparison among Components of Hadoop

Name	HBase	Hive	MongoDB	Redis	Cassandra	Drizzle

Description	Wide-column store based on Apache Hadoop and on concepts of Big Table	Data Warehouse Software for Querying and Managing Large Distributed Datasets, built on Hadoop	One of the most popular Document Stores	In-memory Database with configurable options performance vs. persistency	Wide-column store based on ideas of BigTable and DynamoDB	MySQL fork with a pluggable micro-kernel and with an emphasis of performance over compatibility
Implementation language	Java	Java	C++	C	Java	C++
Database Model	Wide Column Store	Relational DBMS	Document Store	Key – Value Store	Wide Column Store	Relational DBMS
Consistency Concepts	Immediate Consistency	Eventual Consistency	Eventual Consistency, Immediate Consistency	-	Eventual Consistency, Immediate Consistency	-
Concurrency	Yes	Yes	Yes	Yes	Yes	Yes
Durability	Yes	Yes	Yes	Yes	Yes	Yes
Replication Method	Selected Replication factor	Selected Replication factor	Master – Slave Replication	Master – Slave Replication	Selected Replication factor	Master – Master Replication, Master – Slave Replication

5. Conclusion

We have entered an era of Big Data. The paper describes the concept of Big Data along with 3 Vs, Volume, Velocity and variety of Big Data. The paper also focuses on Big Data processing problems. These technical challenges must be addressed for efficient and fast processing of Big Data. The challenges include not just the obvious issues of scale, but also heterogeneity, lack of structure, error-handling, privacy, timeliness, provenance, and visualization, at all stages of the analysis pipeline from data acquisition to result interpretation. These technical challenges are common across a large variety of application domains, and therefore not cost-effective to address in the context of one domain alone. The paper describes Hadoop which is an open source software used for processing of Big Data.

REFERENCES

[1] S.Vikram Phaneendra & E.Madhusudhan Reddy **“Big Data- solutions for RDBMS problems- A survey”** In 12th IEEE/IFIP Network Operations & Management Symposium (NOMS 2010) (Osaka, Japan, Apr 19{23 2013).

[2] Kiran kumara Reddi & Dnvsl Indira **“Different Technique to Transfer Big Data : survey”** IEEE Transactions on 52(8) (Aug.2013) 2348 { 2355}

[3] Jimmy Lin “**MapReduce Is Good Enough**?” The control project. IEEE Computer 32 (2013).

[4] Umasri.M.L, Shyamalagowri.D ,Suresh Kumar.S **“Mining Big Data:- Current status and forecast to the future”** Volume 4, Issue 1, January 2014 ISSN: 2277 128X

[5] Albert Bifet **“Mining Big Data In Real Time”** Informatica 37 (2013) 15–20 DEC 2012

[6] Bernice Purcell **“The emergence of “big data” technology and analytics”** Journal of Technology Research 2013.

[7] Sameer Agarwal†, Barzan MozafariX, Aurojit Panda†, Henry Milner†, Samuel MaddenX, Ion Stoica “**BlinkDB: Queries with Bounded Errors and Bounded Response Times on Very Large Data**” Copyright © 2013ì ACM 978-1-4503-1994 2/13/04

[8] Yingyi Bu _ Bill Howe _ Magdalena Balazinska _ Michael D. Ernst **“The HaLoop Approach to Large-Scale Iterative Data Analysis”** VLDB 2010 paper “HaLoop: Efficient Iterative Data Processing on Large Clusters.

[9] Shadi Ibrahim⋆ _ Hai Jin _ Lu Lu **“Handling Partitioning Skew in MapReduce using *LEEN*”** ACM 51 (2008) 107–113

[10] Kenn Slagter · Ching-Hsien Hsu **“An improved partitioning mechanism for optimizing massive data analysis using MapReduce”** Published online: 11 April 2013

© Springer Science+Business Media New York 2013.

[11] Ahmed Eldawy, Mohamed F. Mokbel "**A Demonstration of SpatialHadoop:An Efficient MapReduce Framework for Spatial Data**" *Proceedings of the VLDB Endowment, Vol. 6, No. 12 Copyright 2013 VLDB Endowment 21508097/13/10.*

[12] Jeffrey Dean and Sanjay Ghemawat "**MapReduce: Simplified Data Processing on Large Clusters**" **OSDI 2010**

[13] Niketan Pansare1, Vinayak Borkar2, Chris Jermaine1, Tyson Condie "**Online Aggregation for Large MapReduce Jobs**" August 29September 3, 2011, Seattle, WA Copyright 2011 VLDB Endowment, ACM

[14] Tyson Condie, Neil Conway, Peter Alvaro, Joseph M. Hellerstein "**Online Aggregation and Continuous Query support in MapReduce**" *SIGMOD'10,* June 6–11, 2010, Indianapolis, Indiana, USA. Copyright 2010 ACM 978-1-4503-0032-2/10/06.

[15] Jonathan Paul Olmsted **"Scaling at Scale: Ideal Point Estimation with 'Big-Data**" Princeton Institute for Computational Science and Engineering 2014.

[16] Jonathan Stuart Ward and Adam Barker "**Undefined By Data: A Survey of Big Data Definitions"** Stamford, CT: Gartner, 2012.

[17] Balaji Palanisamy, Member, IEEE, Aameek Singh, Member, IEEE Ling Liu, Senior Member, IEEE" **Cost-effective Resource Provisioning for MapReduce in a Cloud**"gartner report 2010, 25

[18] Mrigank Mridul, Akashdeep Khajuria, Snehasish Dutta, Kumar N **" Analysis of Bidgata using Apache Hadoop and Map Reduce"** Volume 4, Issue 5, May 2014" 27

[19] Kyong-Ha Lee Hyunsik Choi **"Parallel Data Processing with MapReduce: A Survey"** SIGMOD Record, December 2011 (Vol. 40, No. 4)

[20] Chen He Ying Lu David Swanson **"Matchmaking: A New MapReduce Scheduling**" in 10th IEEE International Conference on Computer and Information Technology (CIT'10), pp. 2736–2743, 2010