International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume 4, Issue 6 (Nov-Dec 2016), PP.78-84

A Survey on Emerging Trends in Hadoop

[1]Soumya.K, [2]ArunKumar.M.S.
[1]M.Tech Scholar ICET, Mulavoor, [2]Asst.Professor,ICET,Mulavoor
[1] soumyakonline@gmail.com

***Abstract*— Due to the advent of new technologies, devices and communication means like social networking sites, the amount of data produced by mankind is growing rapidly every year. The amount of data produced us from the beginning of time till 2003 was 5 billion gigabytes."90% of the world's data was generated in last few years". Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage. This paper conducts a detailed survey on existing technologies and emerging trends in hadoop for fast processing of petabytes of data within seconds. The technology is changing day by day. New trends in hadoop is popping up fastly.**

***Keywords*— Big data, Hadoop, HDFS, MapReduce, Pig, Hive, Hbase, Spark, Tachyon, SSD.**

I. INTRODUCTION

Big Data is a collection of large datasets that cannot be processed using traditional computing techniques.Traditional database system find difficulties to process these data.Because bigdata is heterogeneous.It involves many areas of business and technology. Big data involves the data produced by different devices and applications.[1]Examples of bigdata are Black Box Data,. Social Media Data,Stock ExchangeData,Power Grid Data,Transport Data, Search Engine Data etc. Thus Big Data includes huge volume, high velocity, and extensible variety of data. The data in it will be of three types.

Structured data: Relational data.

Semi Structured data: XML data. Unstructured data: Word, PDF, Text, Media Logs.[3]

A. Benefits of Big Data

Benefits of bigdata lies in the analysis of social media like facebook.Using the information the marketing agencies are able to learn about the response for their campaigns, promotions, and other advertising mediums. Using the information marketing agencies can categorize their consumers, products and also analyze bsiness strategies which help them to grow in their field.

Bigdata analysis can also be a boon to medical field too.Using the data regarding the previous medical history of patients, hospitals are providing better and quick service.

B. 3 Vs of Big Data

Big data can be analyzed mainly with 3 'V's.analysts and practioners are considering that a data can be called as Bigdata when the its size ranges from 30-50 gigabytes and more. The three 'V's are the following.

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.[1]

There are two otheradditional dimensions can be added as described in [2]- Variability and Complexity

C. Challenges in BigData

The major challenges associated with big data are as follows:

Capturing, data Curation, Storage Searching, Sharing, Transfer, Analysis and Presentation To fulfill the above challenges, organizations normally take the help of enterprise servers[3]

D. Traditional Enterprise Approach

An enterprise will have a computer for storage and process big data. For storage purpose, they seek the help of database vendors such as Oracle, IBM, etc.Here, the user interacts with the application, which in turn handles the part of data storage and analysis.

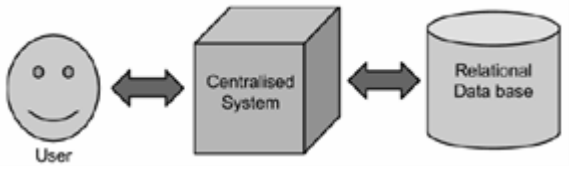

Figure 1: Traditional Enterprise Approach

E. Limitation

This approach works fine with those applications that process less voluminous data and the dta that has a specific structure too. But when it comes to dealing with huge amounts of scalable data, it is difficult to process such data through a single database bottleneck.

F. Google's Solution

Google solved this problem using an algorithm called MapReduce. This algorithm divides the task into small parts and assigns them to many computers, and collects the results from them.[3].

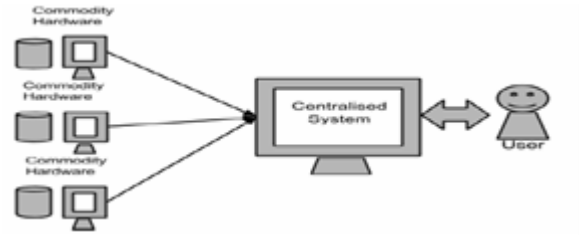

Figure 2: Google's Solution

II. Hadoop

Using the solution provided by Google, Doug Cutting and his team developed an Open Source Project called HADOOP.

Hadoop runs applications using the MapReduce algorithm, where the data is processed in parallel with others.

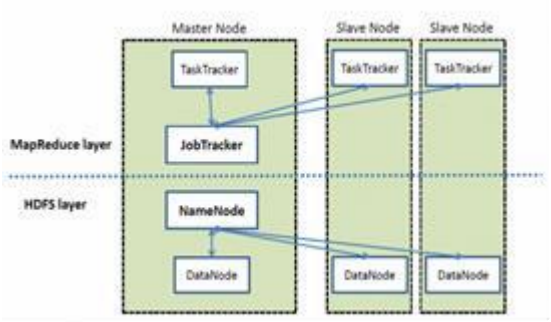

Figure 3: Hadoop Architecture

Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.[1]

A. Hadoop Architecture

At its core, Hadoop has two major layers namely:
(a)Processing layer(MapReduce).
(b) Storage layer (Hadoop Distributed File System).

B. MapReduce

MapReduce was introduced to solve large-data computational problems, and is specifically designedto run on commodity hardware.It can be called as the processing pillar of hadoop. It is based on divide-and-conquer principles — the input data sets are split into independent chunks, which are processed by the mappers in parallel. Additionally,execution of the maps is typically co-locate with the data. The framework then sorts the outputs of the maps, and uses them as an input to the reducers.

MapReduce runs in cluster of nodes;It has a master slave architecture. One node acts as a master node and other nodes act as workers. Workers nodes are responsible for running map and reduce tasks; the master is responsible for assigning tasks to idle workers. Each map worker reads the content of its associated split and extracts key/value pairs and passes it to the user defined Map function. The output of Map function is buffered in memory and partitioned into a set of partitions equals to number ofreducers.

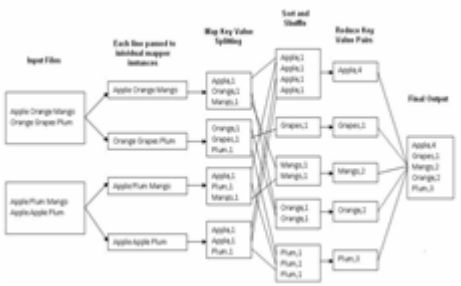

Figure 4: Wordcount Example

C. Mapreduce Execution

First, it localizes the job jar file by copying it to the TaskTracker's filesystem. It also copies any files needed by the application to be on the local disk, and creates an instance of the task runner to run the task. The task runner launches from the distributed cache a new Java virtual machine (JVM) for task execution. The child process (task execution) communicates with its parent (TaskTracker) through the umbilical interface. This way, it informs the parent of the task's progress every few seconds until the task is complete. When the JobTracker receives a notification that the last task for a job is complete, it changes the status for the job to "completed." The job client discovers job completion by periodically polling for the job'sstatus.[4].

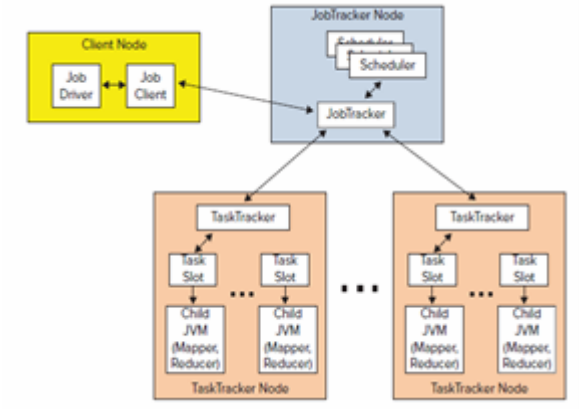

Figure 5: Mapreduce Execution

III. Hadoop Distributed File System

The Hadoop Distributed File System (HDFS) is a distributed file system. HDFS is highly fault-tolerant.. HDFS relaxes a few POSIX requirements to enable streaming access to file system data. HDFS was originally built as infrastructure for the Apache Nutch web search engine project. HDFS is now an Apache Hadoop subproject.

A. HDFS Architecture

HDFS has a master/slave architecture. An HDFS cluster consists of a single NameNode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of

DataNodes. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

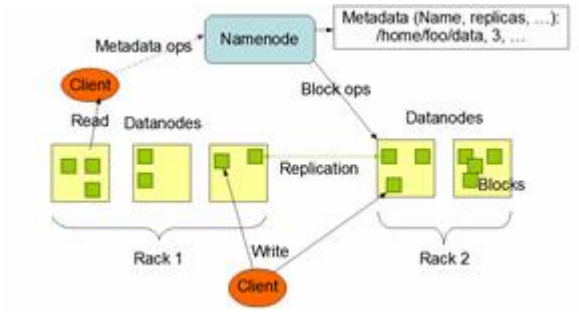

Figure 6: HDFS Architecture

B. The File System Namespace

HDFS supports a traditional hierarchical file organization. A user or an application can create directories and store files inside these directories. The file system namespace hierarchy is similar to most other existing file systems; one can create and remove files, move a file from one directory to another, or rename a file. HDFS does not yet implement user quotas. HDFS does not support hard links or soft links. However, the HDFS architecture does not preclude implementing these features.[5]

An HDFS cluster is comprised of a NameNode, which manages the cluster metadata, and DataNodes that store the data. Files and directories are represented on the NameNode by inodes. Inodes record attributes like permissions, modification and access times, or namespace and disk space quotas.

The file content is split into large blocks (typically 128 megabytes), and each block of the file is independently replicated at multiple DataNodes. The blocks are stored on the local file system on the DataNodes.The Namenode actively monitors the number of replicas of a block. When a replica of a block is lost due to a DataNode failure or disk failure, the NameNode creates another replica of the block.

The NameNode maintains the namespace tree and the mapping of blocks to DataNodes, holding the entire namespace image in RAM.The NameNode does not directly send requests to DataNodes. It sends instructions to the DataNodes by replying to heartbeats sent by those DataNodes. The instructions include commands to:replicate blocks to other nodes, remove local block replicas, re-register and send an immediate block report, or shut down the node.

C. Hive

Hive provides an SQL dialect, called Hive Query Language(abbreviated HiveQL or just HQL) for querying data stored in a Hadoop cluster. Hive is most suited for data warehouse applications, where relatively static data is analyzed,fast response times are not required, and when the data is not changing rapidly. Hive is not a full database. The design constraints and limitations of Hadoop and HDFS impose limits on what Hive can do. The biggest limitation is that Hive does not provide record-level update, insert, nor delete. Also, because Hadoop is a batch-oriented system, Hive queries have higher latency, due to the start-up overhead for MapReduce jobs. Queries that would finish in seconds for a traditional database take longer for Hive, even for relatively small data sets.Finally, Hive does not provide transactions.

D. Pig

Pig provides an engine for executing data flows in parallel on Hadoop. It includes a language, Pig Latin, for expressing these data flows. Pig Latin includes operators for many of the traditional data operations (join, sort, filter, etc.), as well as the ability for users to develop their own functions for reading, processing, and writing data.

Pig is an Apache open source project. This means users are free to download it as source or binary, use it for themselves, contribute to it, and—under the terms of the Apache License—use it in their products and change it as they see fit.

Pig runs on Hadoop. It makes use of both the Hadoop Distributed File System, HDFS, and Hadoop's processing system, MapReduce.HDFS is a distributed filesystem that stores files across all of the nodes in a Hadoop cluster. It handles breaking the files into large blocks and distributing them across different machines, including making multiple copies of each block so that if any one machine fails no data is lost. It presents a POSIX-like interface to users. By default, Pig reads input files from HDFS, uses HDFS to store intermediate data between MapReduce jobs, and writes its output to HDFS

E. Related Works

So many related works and discussions have been evolved about Hadoop and its enhancements.This section covers a literature survey done on Hadoop.

In [6] the disadvantage of existing HDFS is scalability of Namenode.When the usage of memory meets its maximum,the Namenode will not respond.So this paper suggest usage of multiple name nodes.So if one name node fails we can continue the processing with other namenodes.

Map reduce is having limitations as stated in [9].They were not good for interactive and iterative programs.This paper illustrate about the concept called MRlite.It consist of master slave architecture.Master controls parallel execution.It has NFS server.Master accepts the data from client and distribute to slaves.

Hadoop enhancement can be done using a cache,describe in [10].The proposed system uses Locality Sensitive Hashing(LSH).here a cached data is stored in the distributed file system.Map phase wil execute the file system.Then the result is stored in the cache.

It contains five modules like:

Preprocessing files,Creating file vector,Create signature,then using LSH to find the nearest neighbor.Preprocessing will remove all the stop words and the signature will be generated based on TFID(Term Frequency Inverse Document Frequencies)

Then by using LSH they will find the related files.It take less time to the processing the files.

In [11],[12],[13]SSD and HDD are compared and they have mentioned that SSD increases the throughput and consume less

power. They have done experiments with DFSIO benchmark ,Join, Random Writer etc. They conclude SSD perform better than HDD. Hibench suit is also used.

In [14] the possibilities and challenges of implementing SSD in Hadoop frame work is depicted.While SSD is implemented so many challenges are emerged. Data representation discrepancy, Data Split problem, since SSD will not mive data ,It is performing in-memory computing. There occurs system interface discrepancy. SSD cannot directly contact with the host Hadoop system. They introduced File to LBA converter to solve this issues.

In [15] the concept of Apache Spark is introduced. It is a general purpose cluster computing engine provides application programs interfaces in various programming languages like Java, Scala, Python etc.

Spark supports in-memory computing .It makes processing much faster. Spark uses RDD.It doesnot need to store any intermediary result.Spark uses the concept of lineage.

In [16],focus is given to those that reuse a working set of data across multiple parallel execution.Spark can be mainly used for two usecases:First one is or iterative jobs secondly for interactive jobs,where MapReduce is a failure.The main abstraction of Spark is that it uses RDD(Resilient Distributed Dataset.It is readonly collection of objects partitioned across a set of machines.RDD achieves fault tolerance by lineage.It can be implemented in Scala.

In [17]Apache Spark and MapReduce is compared and evaluated.The differences are tabulated below.

	MapReduce	**Spark**
Map phase	Results are stored in circular buffer	Result is stored to OS buffer cache
	Spills to disk when buffer spills 80%	Os decide whether to spill or not
	All spill files are combined to single file.	Doesnot merge spill files.
Reduce Phase	Intermediate files are pulled by reducer to memory	Map phase pushes data in the form of intermediate file to reducer
	Data spilled to disk s merged to big file and reducer is initiated.	Files are written to reducer memory and reducer is invoked.

Table 1:Difference Between MapReduce and Spark

Spark can be integrated with hadoop.It can be work on top of hadoop ,or as standalone.

Spark canno t replace hadoop framework but it can be integrated with hadoop for better perfornmance.

Emerging Trends In Hadoop

F. Project Myriad

Developers from Ebay, MapR and Mesosphere have collaborated to release Project Myriad, a framework that integrates YARN with Apache Mesos—another open-source cluster manager—to run Big Data workloads on the same clusters as other applications in the datacenter and the cloud. Today its developers submitted Myriad to the Apache Incubator, affirming their commitment to open-source collaboration.

Myriad addresses two important goals for ops. One is improving cluster utilization—rather than the Hadoop cluster crunching numbers overnight and sitting relatively idle during peak web traffic hours, Myriad enables Mesos to dynamically share resources between the Hadoop cluster and Web servers and other applications on demand, even simultaneously co-locating Hadoop jobs on the same machines as other tasks, an approach that can easily double or triple utilization.[7]

With Hadoop hitting mainstream IT with a vengeance, open source projects related to Hadoop are popping up everywhere. Here are the top ten most interesting emerging Hadoop projects for you to keep your eye on. Some of them could well stagnate and die quietly if a superior replacement were to come along, but most of these evolutionary specimens will probably become standard components in most Hadoop distributions.

This list focuses on the Apache community projects because this ecosystem has been the one where the majority of the existing mainstream Hadoop projects are developed and maintained. Also, Apache projects have solid governance criteria that foster an open development process where contributions from its members are judged on their technical merit rather than on a corporate agenda.

G. Accumulo

Apache Accumulo is a data storage project for Hadoop, originally developed by the National Security Agency (NSA) of the United States government. Accumulo is a BigTable implementation for Hadoop. Specifically, Accumulo is a multidimensional sorted map, where each row has a unique key, the rows are stored in sorted order based on this key, and each row can have multiple versions (in other words, dimensions).

There was much interest in the NSA in using HBase as a large-scale data store, but it didn't meet the NSA's internal security requirements. NSA engineers then built Accumulo as their own BigTable implementation and later contributed it to the Apache community. The Accumulo project has since grown an active development community, with contributors from a number of different organizations — not just NSA types, in other words. Accumulo, now supported by a number of major Hadoop vendors, is seeing an increasing adoption rate.

The major feature distinguishing Accumulo from other BigTable implementations is cell-based security, which ensures that only authorized users can see the data stored in any

queried rows. This is implemented by way of security labels, which are stored with each row.

H. Drill

A number of emerging and competing technologies are out there trying to solve the SQL-on-Hadoop problem. Though most of these technologies are single-company solutions, some of them are community driven, with Hive the most prominent example. Apache Drill is inspired by the Google Dremel paper, which presents a design for an interactive system that can query data stored in a distributed file system like HDFS and not have to rely on MapReduce. The design goal for Drill is to be able to scale out to thousands of servers and provide subminute response times for queries operating against petabyte-scale data.

I. Falcon

With the increased integration of Hadoop in data warehousing environments, the industry is seeing a significant need for data integration and governance capabilities in Hadoop. Current approaches for integrating data and meeting governance criteria involve these two choices:

- Buy such tooling from established vendors like IBM and Informatica.
- Write extensive libraries of custom code.

This is what the Apache Falcon project is aiming to address with a set of data management services built specifically for Hadoop. Like Drill, Falcon is an Apache incubator project.

The data management services in Falcon are focused primarily on managing data movement and data transformation. If you aren't familiar with managing data between transactional databases and warehouse databases, this process of data movement and transformation is commonly known as Extract, Transform, and Load (ETL). As part of the framework for handling ETL processes, Falcon includes the ability to store metadata for the data as it is passed through the various ETL stages. Falcon can then provide services for data lifecycle management (for example, executing retention policies), data replication, and tracking data lineage.

J. Giraph

Hadoop is quite good at storing and processing data in traditional tables (Hive) and in the newer, BigTable style (HBase and Accumulo), but in many cases, alternative data storage structures are more suited to the task at hand. Graph data looks quite different from table data: It has no rows or columns. There is simply a graph, where individual nodes (also known as vertices) are connected to each other by edges.

Think about it: One huge technical challenges that Google faces is figuring out how best to calculate the ranking of search results. One factor in this is determining how popular individual web pages are, based on how many inbound links originate from other web pages. By far the most practical way to calculate this for all pages is to represent the entire World Wide Web as a graph, where the pages are the nodes and the links are the vertices. To capture its graph database work, Google published a paper on its graph database, which is named Pregel.

Apache Giraph, a graph processing engine that is based on the Pregel paper and was built specifically for Hadoop, can read and write data from a number of standard Hadoop sources, including Hive, HBase, and Accumulo.

The Giraph community is quite large and diverse, featuring code committers from a number of organizations, including Facebook, Twitter, and LinkedIn. Giraph is firmly established as the leading graph processing engine for Hadoop, in terms of code maturity, performance, and adoption. Major Hadoop vendors are now supporting Giraph and will likely include it. (The Apache BigTop project already does.)

K. Knox Gateway

As a distributed system with hundreds or thousands of individual computers, Hadoop clusters are a security administrator's nightmare. To make matters worse, the compute nodes in a Hadoop cluster all have multiple services that talk to each other and, in some cases, require direct connectivity with client applications. Add up all these factors and you have a massive surface area of computers with open ports that you need to protect. To solve this problem, Hortonworks has started the Apache Knox Gateway project, which is still in its early days as an Apache incubator project. The main objective of Knox Gateway is to provide perimeter security for Hadoop clusters. It accomplishes this by providing a central point for cluster authentication on the edge of a Hadoop cluster. Outside the cluster itself, Knox Gateway handles all inbound client requests to a cluster it's guarding and then routes valid requests to the appropriate service in the Hadoop cluster. In this sense, Knox Gateway is literally a secure gateway for all communications between the Hadoop cluster and the outside world. Knox Gateway allows network administrators to isolate the Hadoop cluster from the outside world, because as long as the Knox Gateway servers are active, clients have a secure connection to their Hadoop services.

L. Samza

One exciting aspect of YARN is the possibility of running different kinds of workloads on a Hadoop cluster. With MapReduce, you're restricted to batch processing, but with new technologies such as Spark and Tez (which we talk about below) and the aforementioned Drill, Hadoop will be able to support interactive queries as well. Another class of workload is streaming data, which is what the Apache Samza project is aiming to tackle. (Streaming data works to handle data in real time instead of relying on the stop-and-go aspect of batch processing.)

The Samza project was started by engineers from LinkedIn, which needed a streaming data engine. Rather than keep their code in-house, LinkedIn engineers are developing Samza in the open source Apache community. At the time of this writing, Samza is still in its early days as an Apache incubator project. Though other streaming data engines exist (such as Spark Streaming and Storm, discussed below), the LinkedIn team decided to build its own engine that would best suit its needs.

M. Sentry

The section about the Knox Gateway project above features some of the security challenges with Hadoop. Though Knox Gateway addresses system authorization (ensuring that users are allowed to connect to the Hadoop cluster's services), it doesn't address the pressing need of data authorization, where

there are business needs for restricting access to subsets of data. A common example is the need to hide tables that contain sensitive data such as credit card numbers from analysts looking for behavior patterns. The Apache Sentry project was started by Cloudera as a way to provide this kind of access control to data stored in its Impala project and in Hive. As of Spring 2014, Sentry is an Apache incubator project.

Sentry introduces the concept of different user role classes to Hadoop while enabling the classification of data assets in Impala or Hive. Depending on the classification that's applied at the database, table, or view level, only users with the appropriate roles would be able to access data.

N. Storm

Apache Storm is the third streaming data analysis engine covered in this article (with Samza and Spark Streaming as the other two), which is a testament to how much attention real-time analytics is getting in the Hadoop community. But these divergent approaches are also indications that it's still early in the evolution of streaming data analysis on Hadoop, because none of these three has emerged as a leader. Storm has been an active project the longest, having been donated to the open source community after being acquired by Twitter in 2011. Storm is now an Apache incubator project.

Thanks to work by Hortonworks developers who brought it into the Apache community, Storm was retrofitted to work with the YARN framework. This brought Storm into the Hadoop ecosystem as a real-time processing alternative.

O. Tez

Similar to what is happening with streaming data analysis engines, a number of alternatives have emerged with MapReduce for interactive distributed processing. Spark is a prominent example of these frameworks. The other leading example of such a framework is Apache Tez, which is largely driven by Hortonworks.

The Hortonworks solution to the SQL-on-Hadoop challenge is to improve Hive. To meet this challenge, Hortonworks announced its Stinger initiative, which involved a number of changes to Hive, involving better support for the ANSI SQL standards and much improved performance. One key limitation in Hive is its dependence on MapReduce for processing queries. MapReduce is limited in its ability to deal with common SQL operations such as joins and group-bys, which results in extremely poor performance compared to the massively parallel relational database alternatives running at a comparable large scale. Hortonworks announced the Tez project to present an alternative framework to MapReduce, which is optimized for more optimal (and flexible) data processing possibilities. Tez will also be used as the underlying framework for Pig.

P. Tachyon

Tachyon, an in-memory file system with OrangeFS,a parallel file system[31].It is a reliable data sharing at memory-speed within and across cluster frameworks/jobs.Tachyon enables avoiding GC overhead fine-grained executors fast RDD sharing High-throughput, fault-tolerant in-memory storage Interface compatible to HDFSFurther improve performance for Spark, Shark, and Hadoop .Tachyon's vision is to use memory aggressively and push the lineage to storage layer.One in-memory copy.Rely on recomputation for fault-tolerance .

CONCLUSIONS

This paper includes the bigdata processing framework Hadoop .Here covers a wide technologies which is emerging in the field of Hadoop.Spark integration is also discussed.Limitation of MapReduce is also seen.This paper can be concluded by pointing that spark can be integrated with hadoop also so many emerging technologies are popping up.This survey can be helpful to familiarize about various exisiting technologies and emerging trends in Hadoop.

Q. Future Works

Spark can be used for two scenarios.Iterative and interactive jobs.Future works will be based on the enhancement on Spark.Also various technologies familiarised can be accompanied along this.The scope of SSD is also discussed.So future works will be focusing on the enhancement of Hadop framework by using the emerging technologies.

REFERENCES

[1] Survey Paper On Big Data,Ms. Vibhavari Chavan, Prof. Rajesh. N. Phursule,Department of Computer Engineering,JSPM's Imperial College of Engineering and Research, Pune, Vibhavari Chavan et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (6) , 2014, 7932-7939

[2] A Review Paper on Big Data and Hadoop Harshawardhan S. Bhosale1, Prof. Devendra P. Gadekar, International Journal of Scientific and Research Publications, Volume 4, Issue 10, October 2014 1 ISSN 2250-3153

[3] www.tutorialpoint.com.

[4] Professional Hadoop Solutions, Boris Lublinsky,Kevin T. Smith,Alexey Yakubovich

[5] Apache Hadoop Documentation 2014 http://hadoop.apache.org/.

[6] The Hadoop Distributed File System, Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler.

[7] http://readwrite.com/2015/02/11/project-myriad-big-data-hadoop-yarn-mesos/https://www.oreilly.com/ideas/data-emerging-trends andtechnologies/page/2/ch02

[8] http://www.dummies.com/programming/big-data/hadoop/10-emerging-hadoop-technologies-to-keep-your-eye-on/

[9] The Limitation of MapReduce: A Probing Case and a Lightweight Solution, Zhiqiang Ma Lin Gu ,International Journal of Engineering Trends and Technology (IJETT)2013

[10] Speed-up Extension to Hadoop System, Sayali Ashok Shivarkar, International Journal of Engineering Trends and Technology (IJETT) 2013

[11] A Case for Flash Memory SSD in Hadoop Applications , International Journal of Control and Automation , Seok-Hoon Kang, Dong-Hyun Koo, Woon-Hak Kang and Sang-Won Lee 2013

[12] The Truth About MapReduce Performance on SSDs,Large Installation System Administration Conference (LISA14) , Jaeyoung Do, Yang-Suk Kee, JigneshM.Patel,ChanikPark,KwanghyunPark, David J. DeWitt,2014

[13] How Much Solid State Drive Can Improve the Performance of Hadoop Cluster? Performance evaluation of Hadoop on SSD

and HDD , International Journal of Modern Communication Technologies & Research (IJMCTR), Piyush Saxena, Dr. Jerry Chou,2014.

[14] In-StorageComputing for Hadoop MapReduceFramework: Challenges and Possibilities. IEEE Transactions On Computers, Dongchul Park, Yang-Suk Kee ,2015.

[15] Big Data Analysis: Ap Spark Perspective, Global Journal of Computer Science and Technology: C Software & Data Engineering, Abdul Ghaffar Shoro & Tariq Rahim Soomro ,2015

[16] Spark: Cluster Computing withWorking Sets Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, Ion Stoica University of California, Berkeley,2015

[17] Comparing Apache Spark and Map Reduce with Performance Analysis using K-Means, International Journal of Computer Applications , Satish Gopalani ,Rohan Arora ,2015.

[18] Programming Hive, Edward Capriolo, Dean Wampler, and Jason Rutherglen, Copyright © 2012 Edward Capriolo, Aspect Research Associates, and Jason Rutherglen. All rights reserved.

[19] Learning Spark, Holden Karau, Andy Konwinski, Patrick Wendell, andMatei Zaharia, Copyright c 2015 Databricks. All rights reserved.

[20] Survey on MapReduce Scheduling Algorithms , International Journal of Engineering Trends and Technology (IJETT) Liya Thomas, Syama R, 2014.

[21] Hadoop: The Definitive Guide,4th Edition,Tom White.

[22] Machine Learning, Wikipedia, 2014 http://en.wikipedia.org/wiki/Machine_learning .

[23] Machine learning with Spark - Spark Summit 2013 .

[24] Spark Job Flow – Databricks https://databrickstraining.s3.amazonaws.com/slides/advanced-spark-training.pdf .

[25] Big Data Analytics Hadoop and Spark, Shelly Garion, Ph.D.IBM Research Haifa.

[26] Improving MapReduce Performance in Heterogeneous Environments, Matei Zaharia, Andy Konwinski, Anthony D. Joseph, Randy Katz, Ion Stoica University of California, Berkeley, 8th USENIX Symposium on Operating Systems Design and Implementation

[27] Map-Reduce Implementations: Survey And Performance Comparison, Zeba Khanam and Shafali Agarwal, International Journal of Computer Science & Information Technology (IJCSIT) Vol 7, No 4, August 2015.

[28] Scaling Spark in the Real World:Performance and Usability, Michael Armbrust, Tathagata Das, Aaron Davidson, Ali Ghodsi, Andrew Or,Josh Rosen, Ion Stoica, Patrick Wendell, Reynold Xin, Matei Zaharia.