Impact Factor (SJIF): 3.632

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 3, Issue 4, April -2016

A Review on "How YARN Overcomes MapReduce Limitations in Hadoop 2.0?"

SRI HARI NALLAMALA[1], DR. P. SIVA KUMAR[2], DR. K. SUVARNA VANI[3]

[1]Research Scholar, Dept. of CSE, K L University, Green Fields, Vaddeswaram, Guntur, A.P. &
Assistant Professor, Dept. of CSE, DVR & Dr.HS MIC College of Technology, Kanchikacherla, Krishna Dt., A.P.

[2]Associate. Professor, Dept. of CSE, K L University, Green Fields, Vaddeswaram, Guntur, A.P.

[3]Professor, Dept. of CSE, V.R. Siddhartha Engineering College, Vijayawada, Krishna Dist., A.P.

Abstract:

Though this paper we will discuss about two new labels YARN and MR2 launched in Hadoop 2.0. We will observe, What is YARN? Why there was a need of YARN (Yet Another Resource Negotiator), which a new framework introduced in Hadoop 2.0? What are the benefits associated with YARN framework over earlier MapReduce framework of Hadoop 1.0? Precisely What is the difference between MR1 in Hadoop 1.0 and MR2 in Hadoop2.0? Finally, we concluded both MapReduces.

Keywords: Bigdata, Apache Hadoop, HDFS, MapReduce, YARN, MR1, MR2.

I. INTRODUCTION

You can understand this topics in a better manner if you have basic knowledge of Hadoop and MapReduce. Here a little bit of introduction to Hadoop and MapReduce which will help to us to learn in an efficient manner.

In this paper, we will see the brief descriptions of Hadoop and its related clusters along with their workings.

II. WHAT IS HADOOP?

Hadoop gets a lot of buzz these days in database. This open source software platform managed by the Apache Software Foundation has proven to be very helpful in storing and managing vast amounts of data cheaply and efficiently. But many people in the industry still don't really know what exactly Hadoop is.

A) Hadoop Overview:

Hadoop (also known as Apache Hadoop) is an open source, Java-based programming framework that supports the processing of large data sets in a distributed computing environment. It is part of the Apache project sponsored by the Apache Software Foundation.It is designed to scale up from a single server to thousands of machines, with a very high degree of fault tolerance. Rather than relying on high-end hardware, the resiliency of these clusters comes from the software's ability to detect and handle failures at the application layer.Hadoop is designed to be robust, in that your Big Data applications will continue to run even when individual servers or clusters fail.

B) Hadoop is not a database:

Hadoop an efficient distributed file system and not a database. It is designed specifically for information that comes in many forms, such as server log files or personal productivity documents. Anything that can be stored as a file can be placed in a Hadoop repository.

C) A Brief History of Hadoop:

Hadoop was inspired by Google's MapReduce, a software framework in which an application is broken down into numerous small parts. Any of these parts (also called fragments or blocks) can be run on any node in the cluster. Doug Cutting, Hadoop's creator, named the framework after his child's stuffed toy elephant.

D) What problems can Hadoop solve?

The Hadoop platform was designed to solve problems where you have a lot of data "perhaps a mixture of complex and structured data" and it doesn't fit nicely into tables. It's for situations where you want to run analytics that are deep and computationally extensive, like clustering and targeting. That's exactly what Google was doing when it was indexing the web and examining user behavior to improve performance algorithms.

Hadoop applies to a bunch of markets. In finance, if you want to do accurate portfolio evaluation and risk analysis, you can build sophisticated models that are hard to jam into a database engine. But Hadoop can handle it. In online retail, if you want to deliver better search answers to your customers so they're more likely to buy the thing you show them, that sort of problem is well addressed by the platform Google built.

Hadoop is used for:

- Search - Yahoo, Amazon, Zvents
- Log processing - Facebook, Yahoo
- Data Warehouse - Facebook, AOL
- Video and Image Analysis - New York Times, Eyealike

E) Hadoop Architecture:

Hadoop is designed to run on a large number of machines that don't share any memory or disks. That means you can buy a whole bunch of commodity servers, slap them in a rack, and run the Hadoop software on each one. When you want to load all of your organization's data into Hadoop, what the software does is bust that data into pieces that it then spreads across your different servers. There's no one place where you go to talk to all of your data; Hadoop keeps track of where the data resides. And because there are multiple copy stores, data stored on a server that goes offline or dies can be automatically replicated from a known good copy.

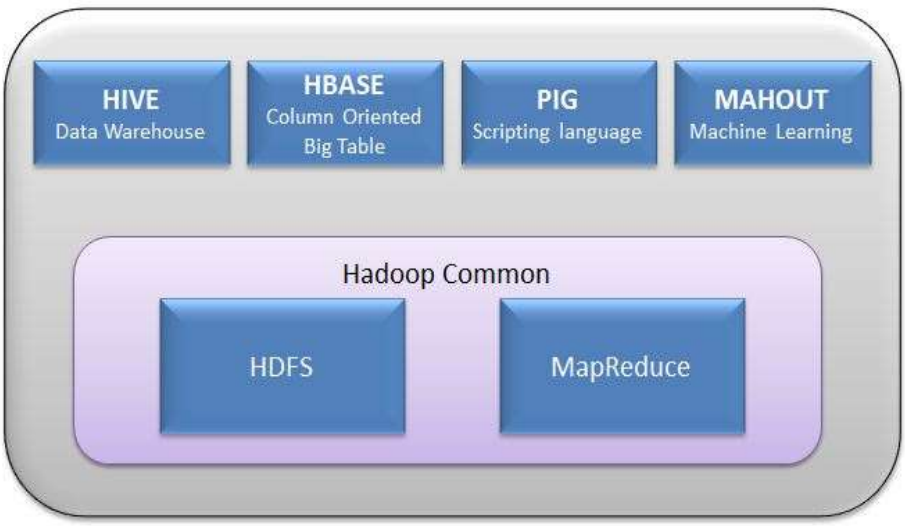
Architecturally, the reason you're able to deal with lots of data is because Hadoop spreads it out. And the reason you're able to ask complicated computational questions is because you've got all of these processors, working in parallel, harnessed together.

Components of Hadoop:

The current Apache Hadoop ecosystem consists of the Hadoop kernel, MapReduce, the Hadoop distributed file system (HDFS) and a number of related projects such as Apache Hive, HBase and Zookeeper. MapReduce and Hadoop distributed file system (HDFS) are the main component of Hadoop.

F) Hadoop distributed file system (HDFS):

HDFS is the file system that spans all the nodes in a Hadoop cluster for data storage. It links together the file systems on many local nodes to make them into one big file system. HDFS assumes nodes will fail, so it achieves reliability by replicating data across multiple nodes.



G) Advantage of Hadoop:

It's Scalable:

New nodes can be added as needed and added without needing to change data formats, how data is loaded, how jobs are written, or the applications on top.

It's Cost effective:

Hadoop brings massively parallel computing to commodity servers. The result is a sizeable decrease in the cost per terabyte of storage, which in turn makes it affordable to model all your data.

It's Flexible:

Hadoop is schema-less, and can absorb any type of data, structured or not, from any number of sources. Data from multiple sources can be joined and aggregated in arbitrary ways enabling deeper analyses than any one system can provide.

It's Fault tolerant:

When you lose a node, the system redirects work to another location of the data and continues processing without missing a beat.

III) MAPREDUCE - THE HEART OF HADOOP

The framework that understands and assigns work to the nodes in a cluster.We will learn about the following:

- What is MapReduce?
- Few interesting facts about MapReduce.
- MapReduce component and architecture.
- How MapReduce works in Hadoop?

A) MapReduce:

MapReduce is a programming model which is used to process large data sets in a batch processing manner.

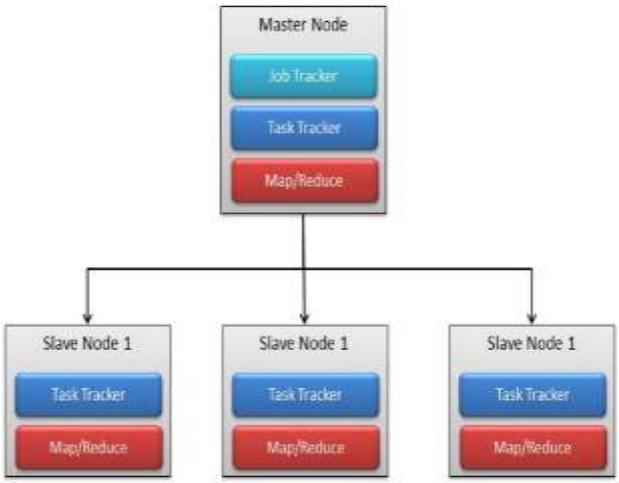
A MapReduce program is composed of

- a Map() procedure that performs filtering and sorting (such as sorting students by first name into queues, one queue for each name)
- And a Reduce() procedure that performs a summary operation (such as counting the number of students in each queue, yielding name frequencies).

Few Important Facts about MapReduce:

- Apache Hadoop Map-Reduce is an open source implementation of Google's Map Reduce Framework.
- Although there are so many map-reduce implementation like Dryad from Microsoft, Dicso from Nokia which have been developed for distributed systems but Hadoop being the most popular among them offering open source implementation of Map-reduce framework.
- Hadoop Map-Reduce framework works on Master/Slave architecture.

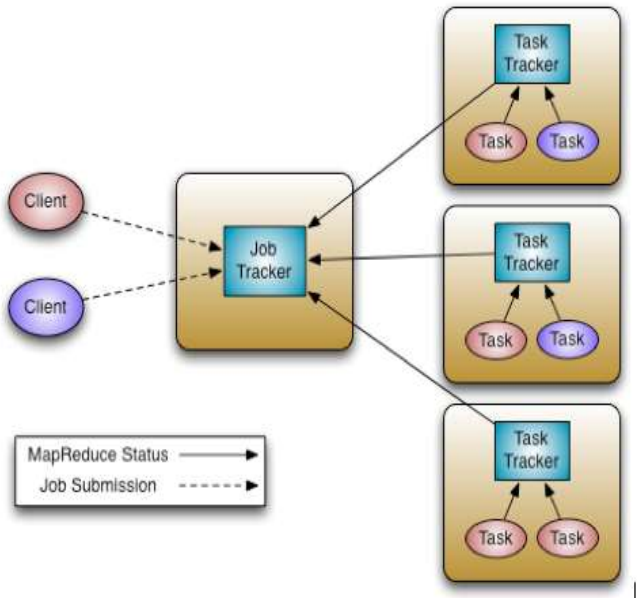
B) MapReduce Architecture:



Hadoop 1.x MapReduce is composed of two components.

1. Job tracker playing the role of master and runs on MasterNode (Namenode)
2. Task tracker playing the role of slave per data node and runs on Datanodes

a) Job Tracker:



1. Job Tracker is the one to which client application submit MapReduce programs (jobs).
2. Job Tracker schedule clients jobs and allocates task to the slave task trackers that are running on individual worker machines (date nodes).
3. Job tracker manage overall execution of Map-Reduce job.
4. Job tracker manages the resources of the cluster like:
    - Manage the data nodes i.e. task tracker.
    - To keep track of the consumed and available resource.
    - To keep track of already running task, to provide fault-tolerance for task etc.

b) Task Tracker:

1. Each Task Tracker is responsible to execute and manage the individual tasks assigned by Job Tracker.
2. Task Tracker also handles the data motion between the map and reduce phases.
3. One Prime responsibility of Task Tracker is to constantly communicate with the Job Tracker the status of the Task.
4. If the JobTracker fails to receive a heartbeat from a TaskTracker within a specified amount of time, it will assume the TaskTracker has crashed and will resubmit the corresponding tasks to other nodes in the cluster.

C) How MapReduce Engine Works:

Let us understand how exactly map reduce program gets executed in Hadoop. What is the relationship between different entities involved in this whole process?

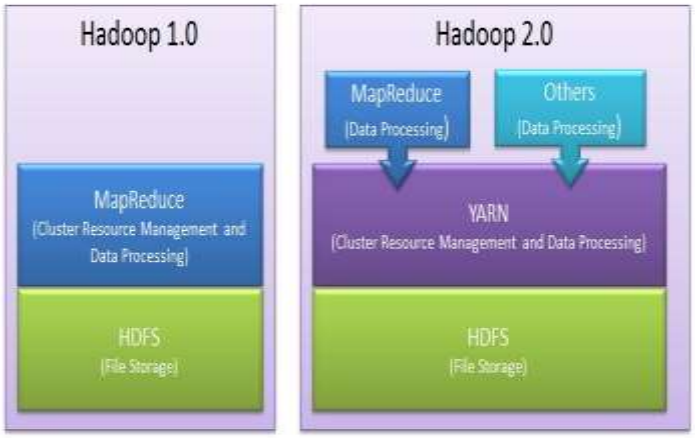
The entire process can be listed as follows:

1. Client applications submit jobs to the JobTracker.
2. The JobTracker talks to the NameNode to determine the location of the data
3. The JobTracker locates TaskTracker nodes with available slots at or near the data

4. The JobTracker submits the work to the chosen TaskTracker nodes.
5. The TaskTracker nodes are monitored. If they do not submit heartbeat signals often enough, they are deemed to have failed and the work is scheduled on a different TaskTracker.
6. A TaskTracker will notify the JobTracker when a task fails. The JobTracker decides what to do then: it may resubmit the job elsewhere, it may mark that specific record as something to avoid, and it may may even blacklist the TaskTracker as unreliable.
7. When the work is completed, the JobTracker updates its status.
8. Client applications can poll the JobTracker for information.

IV) INTRODUCTION OF NEW YARN LAYER IN HADOOP 2.0

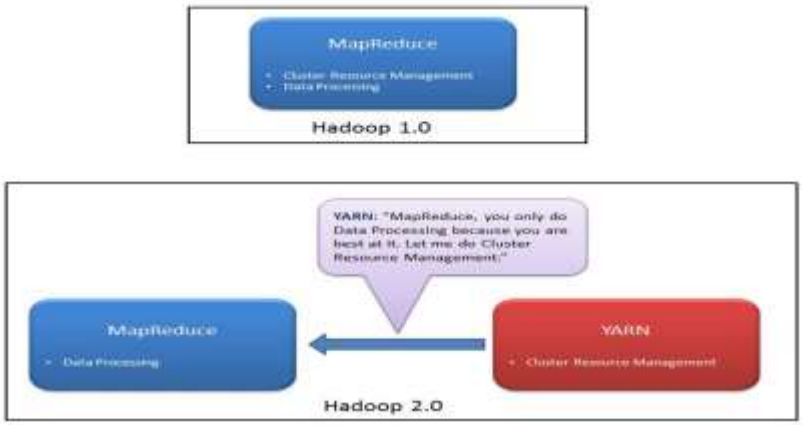
YARN (**Y**et **A**nother **R**esource **N**egotiator) is a new component added in Hadoop 2.0. Now we will see how the Hadoop architecture has changed from Hadoop 1.0 to Hadoop 2.0



As shown, in Hadoop 2.0 a new layer has been introduced between HDFS and MapReduce. This is YARN framework which is responsible for doing Cluster Resource Management.

Cluster resource management means managing the resources of the Hadoop Clusters. And by resources we mean Memory, CPU etc.

YARN took over this task of cluster management from MapReduce and MapReduce is streamlined to perform Data Processing only in which it is best.



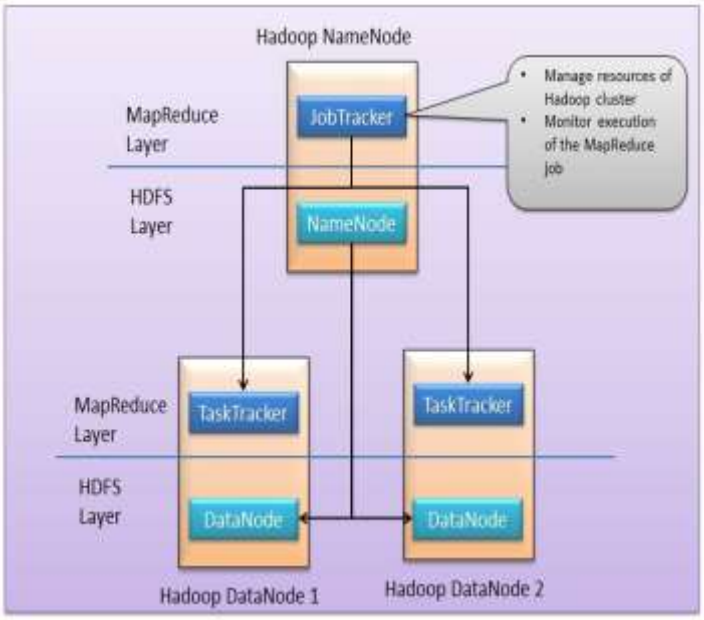
A) Why YARN was needed?

Before we understand the need of YARN, we should understand how cluster resource management was done in Hadoop 1.0 and what the problem in that approach was.

Cluster Resource Management in Hadoop 1.0:

In Hadoop 1.0, there is **tight coupling between Cluster Resource Management and MapReduce programming model**.

Job Tracker, which does resource management, is part of, MapReduce Framework.



In MapReduce framework, MapReduce job (MapReduce application) is divided between number of tasks called mappers and reducers. Each task runs on one of the machine (DataNode) of the cluster, and each machine has a limited number of predefined slots (map slot, reduce slot) for running tasks concurrently.

Here, JobTracker is responsible for both managing the cluster's resources and driving the execution of the MapReduce job. It reserves and schedules slots for all tasks, configures, runs and monitors each task, and if a task fails, it allocates a new slot and reattempts the task. After a task finishes, the job tracker cleans up temporary resources and releases the task's slot to make it available for other jobs.

B) Problems with this approach in Hadoop 1.0:

- **It limits scalability:** JobTracker runs on single machine doing several task like
  - Resource management
  - Job and task scheduling and
  - Monitoring

  Although there are so many machines (DataNode) available; they are not getting used. This limits scalability.
- **Availability Issue:** In Hadoop 1.0, JobTracker is single Point of availability. This means if JobTracker fails, all jobs must restart.
- **Problem with Resource Utilization:** In Hadoop 1.0, there is concept of predefined number of map slots and reduce slots for each TaskTrackers. Resource Utilization issues occur because maps slots might be 'full' while reduce slots is empty (and vice-versa). Here the compute resources (DataNode) could sit idle which are reserved for Reduce slots even when there is immediate need for those resources to be used as Mapper slots.
- **Limitation in running non-MapReduce Application:** In Hadoop 1.0, Job tracker was tightly integrated with MapReduce and only supporting application that obeys MapReduce programming framework can run on Hadoop.

  Let's try to understand this point in more detail.

  Hadoop distributed file system (HDFS) makes it cheap to store large amounts of data, and its scalable MapReduce analysis engine makes it possible to extract insights from that data. MapReduce works on batch-driven data analysis, where the input data is partitioned into smaller batches that can be processed in parallel across many machines in the Hadoop cluster. But MapReduce, while powerful enough to express

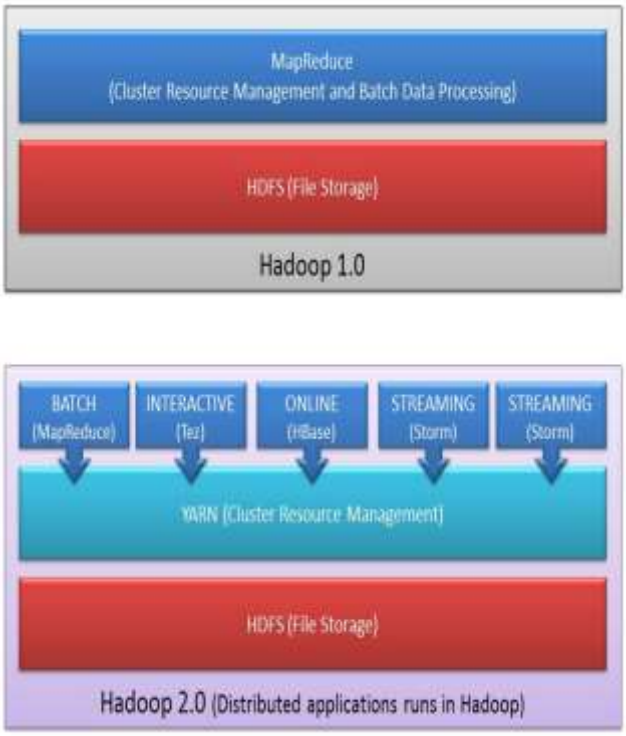
many data analysis algorithms, is not always the optimal choice of programming paradigm. It's often desirable to run other computation paradigms in the Hadoop cluster – here are some examples.

- **Problem in performing real-time analysis:** MapReduce is batch driven. What if I want to do perform real time analysis instead of batch-processing (where results is available after several hours).
  There are many applications which need results in real time like fraud detection algorithm. There are real time engines like Apache Storm which can work better in this case. But in Hadoop 1.0, due to tight coupling these engines cannot run independently.
- **Problem in running Message-Passing approach:** It is a stateful process that runs on each node of a distributed network. The processes communicate with each other by sending messages, and alter their state based on the messages they receive. This is not possible in MapReduce.
- **Problem in running Ad-hoc query:** Many users like to query their big data using SQL. Apache Hive can execute a SQL query as a series of MapReduce jobs, but it has shortcomings in terms of performance.

  Recently, some new approaches such as Apache Tajo, Facebook's Presto and Cloudera's Impala drastically improve the performance, but they require to run services in other form than MapReduce form.

  It is not possible to run all such non Map Reduce jobs on Hadoop Cluster. Such jobs have to "disguise" themselves as mappers and reducers in order to be able to run on Hadoop 1.0.

C) Hadoop 2.0 solves all these problems with YARN:



YARN took over the task of cluster management from MapReduce and MapReduce is streamlined to perform Data Processing only in which it is best.

YARN has central resource manager component which manages resources and allocates the resources to the application. Multiple applications can run on Hadoop via YARN and all application could share common resource management.

Advantages of YARN:

1. **Yarn does efficient utilization of the resource.**
   There are no more fixed map-reduce slots. YARN provides central resource manager. With YARN, you can now run multiple applications in Hadoop, all sharing a common resource.

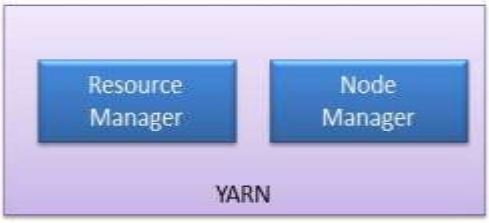
2. **Yarn can even run application that do not follow MapReduce model.**
   YARN decouples MapReduce's resource management and scheduling capabilities from the data processing component, enabling Hadoop to support more varied processing approaches and a broader array of applications. For example, Hadoop clusters can now run interactive querying and streaming data applications simultaneously with MapReduce batch jobs. This also streamlines MapReduce to do what is does best - process data.

Few Important Notes about YARN:

1. **YARN is backward compatible.**
   This means that existing MapReduce job can run on Hadoop 2.0 without any change.

2. **No more JobTracker and TaskTracker needed in Hadoop 2.0**

   JobTracker and TaskTracker has totally disappeared. YARN splits the two major functionalities of the JobTracker i.e. resource management and job scheduling/monitoring into 2 separate daemons (components).
   - Resource Manager
   - Node Manager(node specific)
     Central Resource Manager and node specific Node Manager together constitutes YARN.



V) MAPREDUCE: DIFFERENCE BETWEEN MR1 AND MR2

Earlier version of map- reduce framework in Hadoop 1.0 is called as **MR1**. The new version of MapReduce is known as **MR2**.

No more JobTracker and TaskTracker needed in Hadoop 2. With the introduction of YARN in Hadoop2, the term JobTracker and TaskTracker disappeared. MapReduce is now streamlined to perform processing data.

The new model is more isolated and scalable as compared to the earlier MR1 system. MR2 is one kind of distributed application that run MapReduce framework on top of YARN. MapReduce perform data processing via YARN. Other tools can also perform data processing via YARN. Hence Yarn execution model is more generic than earlier MapReduce model.

MR1 was not able to do so. It would only run MapReduce applications.

VI) CONCLUSION

MapReduce in Hadoop 1.0 – MR1 will use the JobTracker to create and assign tasks to TaskTrackers, which could become a useful resource bottleneck when the cluster scales out far enough (commonly around 4,000 to 5,000 clusters.)

MapReduce in Hadoop 2.0 (YARN) – MR2 has a Resource Manager for each and every cluster, and each data node runs a Node Manager. In YARN, the functions of the JobTracker have been split between three services. The ResourceManager is a persistent YARN service that receives and runs applications (a MapReduce job is an application) on the cluster. YARN contains the scheduler, which, as previously, is pluggable. The MapReduce-specific abilities of the JobTracker have been shifted into the MapReduce Application Master, one of which is started to deal with each MapReduce job and terminated when the job finishes. The JobTracker function of serving information about completed jobs has been sent to the JobHistory Server. The TaskTracker has been replaced with the NodeManager, a YARN service that manages resources and deployment on a host.

So, with this review paper, finally we concluded that YARN i.e. MR2 is best for MapReduce process compared to MR1.

REFERENCES

[1] Ashish Sharma, Snehlata Vyas "Hadoop2 Yarn", IPASJ International Journal of Computer Science (IIJCS), Volume 3, Issue 9, September 2015.

[2] 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.

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

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

[5] 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.

[6] 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.

[7] Hadoop http://hadoop.apache.org & http://sortbenchmark.org/YahooHadoop.pdf

[8] Hadoop. http://hadoop.apache.org, 2009.

[9] HDFS (hadoop distributed file system) architecture. http://hadoop. apache.org/common/docs/current/hdfs design.html, 2009.

[10] R. Abbott and H. Garcia-Molina. Scheduling I/O requests with deadlines: A performance evaluation. In Proceedings of the 11th Real-Time Systems Symposium, pages 113–124, Dec 1990.

[11] G. Candea, N. Polyzotis, and R. Vingralek. A scalable, predictable join operator for highly concurrent data warehouses. In 35th International Conference on Very Large Data Bases (VLDB), 2009.

[12] R.-I. Chang, W.-K. Shih, and R.-C. Chang. Deadline-modificationscan with maximum-scannable-groups for multimedia real-time disk scheduling. In RTSS '98: Proceedings of the IEEE Real-Time Systems Symposium, page 40, Washington, DC, USA, 1998. IEEE Computer Society.

AUTHOR'S BIOGRAPHY

Mr. Sri Hari Nallamala was received B.Tech degree in Computer Science and Engineering from JNT University Engineering College, Hyderabad in 2007 and M.Tech degree in Computer Science Specialization from JNT University Hyderabad in 2010. Now, he was registered and pursuing his research work at K L University, Vijayawada. Totally, he has 5 years of experience in teaching. Presently he was working as an Assistant Professor in DVR & DRHS MIC College of Technology, Kanchikacherla, Krishna Dist., A.P. His interested areas are Data Mining, Cloud Computing, and Internet Technologies.

Dr. P. Siva Kumar was completed his Under Graduate, Post Graduate and Ph.D. in Computer Science Engineering Specialization. He has more than 10 years of rich experience in teaching & research. Presently he was working as Associate Professor of CSE at K L University, Guntur, A.P. His research areas are Data Mining, Computer Networks, and IOTs. He was guiding number of Scholars at several Universities.

Dr. Koneru Suvarna Vani received B.Tech degree in Computer Science and Engineering and M.Tech degree in Computer Science and Engineering and Ph.D. Computer Science from University of Hyderabad. She has more than 15 years of rich experience in teaching & research. Presently she was working as Professor of CSE at V.R. Siddhartha Engineering College, Vijayawada Krishna Dist., A.P. Her research areas are Data Mining, Bio-informatics. At Present she was guiding 1 Scholar at K L University, Guntur and 2 Scholars at JNTUK, Kakinada.