Advances in Computational Sciences and Technology
ISSN 0973-6107 Volume 10, Number 8 (2017) pp. 2317-2325

http://www.ripublication.com

A survey on Static and Dynamic Hadoop Schedulers

Mukesh Singla
Research Scholar, OPJS University
Rajgarh (Sadulpur) Churu, Rajasthan, India.

Abstract

Hadoop is an open source framework which tracks a distributed computing approach. The choice of Hadoop is growing day by day. The capability to deal with enormously large data which is unstructured in nature is truly challenging. But Hadoop is able to tackle all these challenges. This paper describes Hadoop and its architecture in detail. This paper also creates an overview of schedulers used in Hadoop: Static scheduler and Dynamic scheduler. A comparative analysis of various algorithms of static and dynamic scheduling is also presented.

1. INTRODUCTION

Social media is the immense source of data. The ransom amount of data is collected as millions of users update their opinions, reviews every single second. Users express their thoughts using internet which is easily available and accessible all over the world. With each passing day, the numbers of internet users are increasing rapidly. Various forums are available where users can express their opinions. Social media is the main source for people all over the world where people share their thoughts, express their feelings, spread different opinions using internet services. This makes social media platforms more and more popular day by day. Every minute, large quantity of data is collected. This data collected can be structured, semi-structured or unstructured. So there is a need to handle the problem of dealing with the terabytes and petabytes of data. To analyse and handle such form of data, Hadoop framework is used. Hadoop framework is used for reliable, parallel, scalable and distributed computing.

2. Apache Hadoop

Hadoop was originally developed in 2005 by Doug Cutting and Mike Cafarella which was released by Apache Foundation. It follows an open source implementation. . It supports distributed computing framework but it is different from other distributed frameworks in the sense it provides high fault tolerance power which means in case some machine is failed, its backup will always be available in order to avoid disturbance in processing. Hence availability of data is increased. A large set of data is distributed into small chunks of data which is distributed among various nodes. It scales up data from a single machine and distributes it to several machines which offer its own local storage and computation capabilities. This makes Hadoop reliable, efficient and accurate. Hadoop is less vulnerable to let-downs due to high availability of data.[1]

Released by	Apache Software
Initial release	December 10,2011
Stable release	August 25,2016
Operating system	Cross-platform
File system	Distributed file system
Language	Java

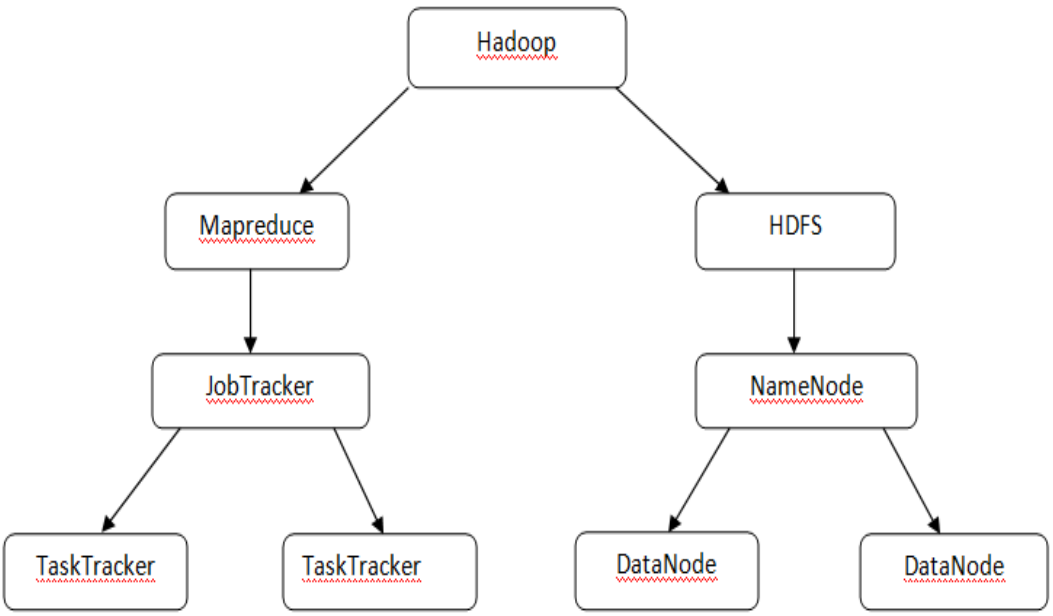

Figure 1.1 Architecture of Hadoop

Hadoop framework includes following modules:[2]

- Hadoop Common:
  Hadoop Common is the base or core of the Hadoop framework. It contains services and processes of the operating system and its file system. It is the collaboration of libraries and common utilities supported by other Hadoop modules. It is also known as Hadoop core.
- Hadoop Distributed File System(HDFS):

HDFS is a handy file system which is written in java. Large files are stored on multiple machines which helps it to achieve reliability.

- MapReduce:
  MapReduce is the operation phase of the large data sets for the Hadoop framework.

2.1 HDFS and its architecture

Hadoop has its own file system which is based on Google File System(GFS). It follows master/slave architecture where master node is the head node which manages the whole file system and the slave is the worker node which provides provisions for storing the actual data. It can admit the data in any format like text files, jpg files, video files, etc. HDFS adds following features to the Hadoop:

- Reliability
- Scalability
- Fault tolerance
- Self-healing file system
- Java based file system

Components of HDFS:

- NameNode
- DataNode
- Secondary NameNode

1. NameNode: NameNode is the metadata server which acts as the master node. It maintains all the files in the file system and retains the record of the location of all the files across the clusters unless it doesn't stores the actual data of the files. It allocates the tasks to the slave DataNode.

   NameNode is considered as the Single point of Failure which means if the NameNode goes down then the whole system goes down. It accomplishes the following function:

- Breaks the file into small blocks.

- Assign the blocks to the DataNode.
- Keeps record of each DataNode and its associated blocks.

2. Secondary NameNode : Secondary NameNode is not a backup for the NameNode in reality but a assistant node for NameNode. The main purpose of the secondary NameNode is to put checkpoints in HDFS.

3. Datanodes: DataNodes act as slave or worker nodes which handles the actual data storage. The data can be in any form, structured or unstructured. They store data blocks and serve all the read and write requests coming from the client.
   Each node has its own DataNode which is controlled by the NameNode. A DataNode periodically sends a heartbeat message to NameNode reporting its status and asks for more information.[3]

2.2 MapReduce and its architecture:

MapReduce is considered as a programming model used in Apache Hadoop which is used for writing applications and processes large sets of in parallel. It processes the data kept in HDFS. It accepts the key/value pair and distributes the key/value pair to each reducer equally.

MapReduce comprises of two phases:

1. Map Phase: The Map phase processes the input data. The input data is in the form of file which is stored in Hadoop's file system. The Mapper function maps the whole input file and divide the data into small chunks of data.
2. Reduce Phase:
   Before Reduce phase, Shuffle phase passes the intermediate results of Map Phase to the Reducer phase. The output of Map phase is passed to the Reducer phase. A new output is created after the whole execution. The final output is stored in the HDFS.[4]
   The job of MapReduce is to divide the input data into small chunks of data, pass these data chunks to the Map phase and the output of the Map phase is treated as the input to the Reduce phase.[3]

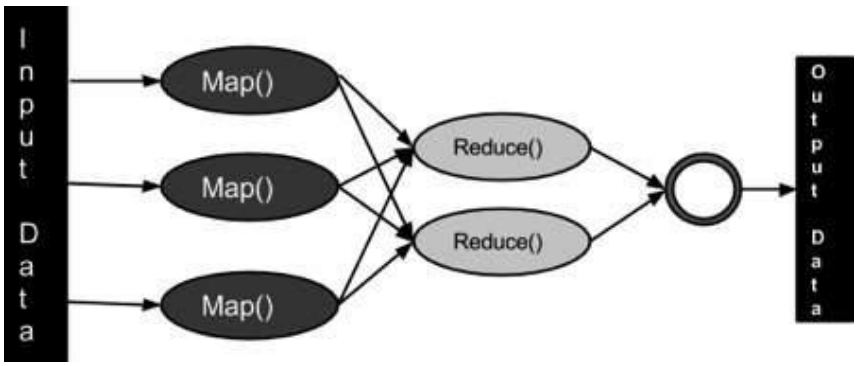

Figure 1.2 Architecture of MapReduce

Components of MapReduce :

- JobTracker
- TaskTracker

1. JobTracker:
   The JobTracker acts as a master node and monitors the MapReduce tasks. It submits the tasks to the slave nodes and asks NameNode to locate the data in HDFS which needs to be processed.
   JobTracker is considered as a single point of failure, means if it goes down, all the executing jobs will came to a halt. When the JobTracker needs specific slots to process a task, it looks for a blank slot on the same server that holds the DataNode which holds the specified data otherwise it looks for an empty slot on a machine having the same rack.
2. TaskTracker:
   The TaskTracker acts as a slave node which accepts jobs from master node and processes the MapReduce tasks. Every TaskTracker has a definite number of set of slots, which indicates the total number of tasks that can be accepted by it. It sends heartbeat messages to Jobtracker periodically to ensure the communication between the two is still alive. It will govern the the failure of a task and will notify the JobTracker about the failure.[3]

2.3 Scheduling in Hadoop:

Scheduling refers to the process of managing and controlling the tasks taking care of every constraint. In Hadoop framework, scheduling is a vital aspect to enhance the overall performance of the Hadoop Cluster. It is responsible for managing multiple nodes in a Hadoop cluster so that all the nodes gets equal share of resources.
There are two types of scheduler:

- Static scheduler
- Dynamic scheduler

In Static scheduling, job is allocated to processors before the execution of task started. Information regarding job execution time and processing resources is known at compile time. The overall execution time of current task is minimized through static scheduling.
In Dynamic scheduling, job is allocated to processors allocation during execution time. When a job begins its execution, a decision is made in the dynamic environment of the system.[6]

3. LITERATURE REVIEW

3.1 Hadoop Architecture:

Mohd Rehan Ghazi[3] in his paper describes that Apache Hadoop consists of two main core components: 1) HDFS, 2) MapReduce. These two modules have five

different daemons and each of them run its own Java Virtual Machine. (1)NameNode, 2)DataNode, 3) Secondary NameNode, 4) JobTracker, 5)TaskTracker

3.2 Scheduling :

Dongjin Yoo[5] in his paper describes that for effective utilization there must be some scheduling mechanism. Multiple jobs in a cluster must be scheduled to improve the utilization of the resources. With scheduling, there are various issues associated. These issues are:

- Locality
- Synchronization
- Fairness

Jyoti V Gautam[6] in his paper describes that job scheduling highly improve the performance of a Hadoop cluster. Hadoop scheduler is classified.

Types of schedulers

- Static schedulers
- Dynamic schedulers

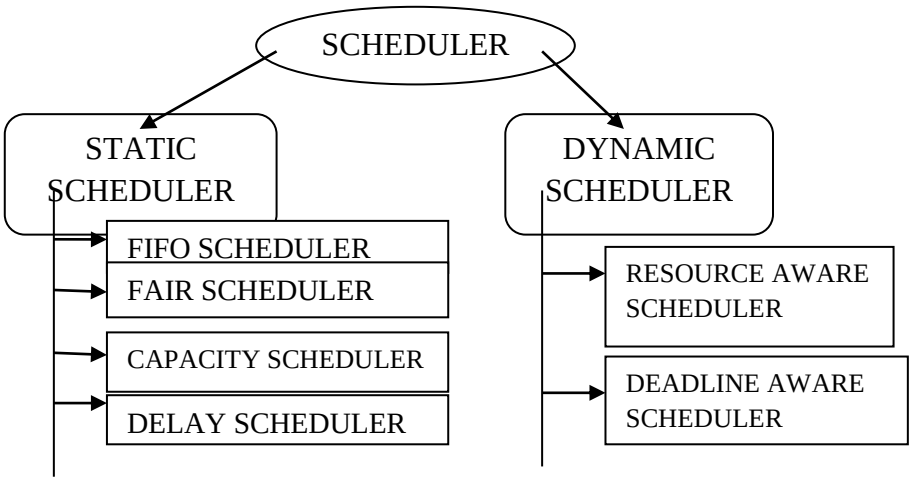

Figure 1.3

Technique Name	Author's name	Description	Scheduler	Environment	Advantages	Disadvantages
Delay Scheduling Based on Histroy TIme	Bin Ye	Slot's performance is evaluated using history time to improve the performance	Static	Heterogeneous	• Improvement in performance of heterogeneous Hadoop system. • More number of jobs are assigned to the slots. • Throughput is	• Performance is better than Fair Scheduler but not satisfactory

		and fairness.			increased.	
Resource and Deadline-Aware Job Scheduling	Dazhao Cheng	Considers future resource availability while deadline of job is minimized.	Dynamic	Heterogeneous	• Resource allocation is optimal • Job deadline misses gets reduced compared to Fair scheduler.	• Availability of dynamic resource is not considered.
Resouce-Aware Adaptive scheduling	Jorda Polo	Adapts itself to changes made in resource demand and availability	Dynamic	Homogeneous	• Resource utilization and job performance is improved.	• Unable to confine the different resource ultization of each jobs in a multiuser environments
Local Resource Shaper for MapReduce	Peng Lu	Uses concept of active and passive slots where active slots can use as many resources as it whereas passive slots can use only those resources which are unused.	Dynamic	Homogeneous	• Improves CPU utilization • Lowers the I/O contention • Reduces job duration.	• Resource usage is fixed so remains fixed by fair resource sharing.
Adaptive Task Scheduling	Xiaolong Xu	This stratergy governs the change in load at runtime. TaskTracker will adapt itself in such scenerios.	Dynamic	Homogeneous + heterogeneous	• Improves the execution efficiency of Hadoop cluster. • Performance is superior. • Execution time of a task is reduced. • Abnormal state is avoided during execution of task	• Not efficient for large scale cluster. • Less reliable and fault tolerant.

4. CONCLUSION

Hadoop has been seen as a technology which has been evolved and improved over years. Hadoop provides an environment for distributed computing. Job scheduling is seen as an important aspect for high performance in Hadoop cluster. This paper reviewed various scheduling algorithms. The schedulers used mainly deal with the resources like CPU cost, memory, I/O. The various scheduling algorithms discussed are classified on the basis of types of scheduler used, the supported environment, advantages and disadvantages of the algorithms. Delay scheduling uses static scheduler which improves the throughput and performance of the Hadoop cluster. While latter all algorithms use dynamic scheduler. Resource aware scheduling optimizes the resource utilization. Local resource shaper improvises the CPU utilization and reduces execution time of the job. Adaptive task scheduling increases the execution efficiency of the Hadoop cluster.

REFERENCE

[1] Hannan SA. An overview on Big Data and Hadoop. International Journal of Computer Applications. 2016 Jan 1;154(10).

[2] Yadav S, Vishwakarma S, Verma A. Efficient & Accurate Scheduling Algorithm for Cloudera Hadoop. In Computational Intelligence and Communication Networks (CICN), 2015 International Conference on 2015 Dec 12 (pp. 839-844). IEEE.

[3] Ghazi MR, Gangodkar D. Hadoop, MapReduce and HDFS: A Developers Perspective. Procedia Computer Science. 2015 Jan 1;48:45-50.

[4] Yoo D, Sim KM. A comparative review of job scheduling for MapReduce. In Cloud Computing and Intelligence Systems (CCIS), 2011 IEEE International Conference on 2011 Sep 15 (pp. 353-358). IEEE.

[5] Gautam JV, Prajapati HB, Dabhi VK, Chaudhary S. A survey on job scheduling algorithms in Big data processing. In Electrical, Computer and Communication Technologies (ICECCT), 2015 IEEE International Conference on 2015 Mar 5 (pp. 1-11). IEEE.

[6] Alam A, Ahmed J. Hadoop architecture and its issues. In Computational Science and Computational Intelligence (CSCI), 2014 International Conference on 2014 Mar 10 (Vol. 2, pp. 288-291). IEEE.

[7] Xu X, Cao L, Wang X. Adaptive task scheduling strategy based on dynamic workload adjustment for heterogeneous Hadoop clusters. IEEE Systems Journal. 2016 Jun;10(2):471-82.

[8] Liu L, Zhou Y, Liu M, Xu G, Chen X, Fan D, Wang Q. Preemptive hadoop jobs scheduling under a deadline. In Semantics, Knowledge and Grids (SKG), 2012 Eighth International Conference on 2012 Oct 22 (pp. 72-79). IEEE.

[9] Cheng D, Rao J, Jiang C, Zhou X. Resource and deadline-aware job

scheduling in dynamic hadoop clusters. In Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International 2015 May 25 (pp. 956-965). IEEE..

[10] Ye B, Dong X, Zheng P, Zhu Z, Liu Q, Wang Z. A delay scheduling algorithm based on history time in heterogeneous environments. In ChinaGrid Annual Conference (ChinaGrid), 2013 8th 2013 Aug 22 (pp. 86-91). IEEE.

[11] Lu P, Lee YC, Gramoli V, Leslie LM, Zomaya AY. Local resource shaper for mapreduce. In Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on 2014 Dec 15 (pp. 483-490). IEEE.

[12] Polo J, Castillo C, Carrera D, Becerra Y, Whalley I, Steinder M, Torres J, Ayguadé E. Resource-aware adaptive scheduling for mapreduce clusters. In ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing 2011 Dec 12 (pp. 187-207). Springer Berlin Heidelberg.