**Job Scheduling in Big Data Processing Using Capacity Scheduler**

**Vignesh Iyer1, Prof. Sweta Bhattacharya2**

*School of Information Technology and Engineering,*

*Vellore Institute of Technology, Vellore-632014*

**Email:-iyer.vignesh2@gmail.com, sweta.b@vit.ac.in**

*.*

**Abstract**

Big data is a popular and a fast progressing domain in Information Technology sector. Big data analytics involves wide range of datasets which are difficult to be managed by existing conventional applications. Big Data analytics has been implemented in many fields such as finance and business, banking, online and onsite purchasing, healthcare, astronomy, oceanography, engineering, and many other fields. Big data a developing field acknowledges many research issues and challenges such as handling data volume, analysis of large amount of data, privacy of data, data visualization, storage of huge amount of data, fault tolerance, development compact data structure for storing heterogeneous data coming from various sources, and job scheduling. Scheduling problem has been an active area of research in computing systems since their inception. Our objective is to study job scheduling algorithms in Big data processing and to propose new job scheduling algorithm in Capacity Scheduler. For job scheduling and data intensive computation, Hadoop software framework is used. Hadoop supports pluggable schedulers and the one widely used in hadoop is Capacity Scheduler. Since it supports pluggable scheduler we can modify the existing algorithm in Capacity Scheduler which will help to reduce the time taken in completing the scheduled jobs in a dynamic environment.

**Keywords**: Big data, Hadoop, MapReduce, Job Scheduling, Classification.

**I. INTRODUCTION**

Big data [3] is one of the most important topic in the field of Information Technology. The reason being it important, is the high demand in maintaining huge amount of datasets generated from many organizations and to maintain them by resolving the issues which are currently faced by the IT sector. Variety of datasets from many fields are being generated from all the conventional companies and are becoming increasingly difficult to handle them with the help of the existing applications. Big data being a popular topic in today’s IT world it is spread in many major high data generating fields such as banking engineering etc. to mention a few. The datasets generated from these fields are huge and sometimes redundant so it becomes very complex to handle them as they grow at a huge rate day by day. The increase in data most importantly depends on three V’s :- Volume, Variety and Velocity. Increase in the amount of data; increase in amount of complexity since they are directly proportional in dynamic environment. The basic steps involved in processing the data are as follows:-1) correlate, 2) link, 3) match, and 4)transform. Performing these steps on Big Data is a very complex and complicated process.

There are many research issues and challenges which are yet to be addressed in Big Data. The main research issues which are to be addressed in big data are follows: 1) Handling a huge amount of data volume, 2) Analyzing big data, 3) Preserving the Privacy of data, 4) Storing huge amount of data, 5) Data visualization, 6) Scheduling the jobs in big data, 7) Fault tolerance.

1) Handling a huge amount of data [1] [2]: To handle huge amount of data coming from a variety of fields biology, astronomy, meteorology, etc is very difficult to compute.

2) Analyzing big data: It is a very difficult process to analyze big data because of the heterogeneous nature of the incoming data and the incompleteness of such data in a dynamic environment. Since these data are available in different formats, in different variety, and different structure [3] analyzing process becomes complex.

3) Preserving the Privacy of data with respect to big data [3]: There is public fear regarding the inappropriate use of personal data, particularly through linking of data from multiple sources. Managing privacy is both a technical and a sociological problem.

4) Storage of huge amount of data [1] [3]: This issue represents the problem of recognizing and storing the most important and critical information which are normally extracted from unstructured data in an efficient manner .

5) Data visualization [1]: To address this issue Data processing techniques should be made efficient enough so that real time visualization is achieved.

6) Job scheduling in big data [8]: This issue mainly focuses on achieving efficient scheduling of various jobs in a distributed and dynamic environment.

7) Fault tolerance [11]: This is one of the major issue in Hadoop framework in the field of big data. Namenode being a single point of failure in Hadoop it becomes very necessary to address this issue. To address the issues of Fault tolerance in Hadoop the techniques to handle them must be efficient enough to handle any kind of failure in distributed and a dynamic environment.

MapReduce [6] an important framework provides an ideal platform for processing such large datasets in various fields generated from big companies by using parallel and distributed programming approaches.

To analyze huge amount of datatsets and to process them we need appropriate tools to carry out the mentioned activities and one of the most widely recognized tool is Apache Hadoop [7]. It is known to handle many types of data:- structured, unstructured, log files, pictures etc. Keeping in mind the dynamic environment, Hadoop also supports redundancy, scalability, parallel processing, and distributed architecture [7]. Many different types of jobs may be available for processing on a Hadoop cluster at any point of time. Appropriate Scheduling policies need to be applied to run MapReduce jobs in a Hadoop cluster comprising of multiple nodes. In this paper we concentrate on Capacity Scheduler which was developed by Yahoo. This scheduler works by naming the queues and to all the queues the jobs are submitted. The queues will be allocated some fraction of the available computing resource. Capacity Scheduler uses two queues namely ready queue and waiting queue. The jobs in waiting queue are assigned priorities. Based on the priority based scheduling the jobs are prioritized in the queue. One of the most important factor on which scheduling is dependent is on workload. The workload in dynamic environment keeps varying all the time in Hadoop. for all kinds of jobs. Because of the various issues involved in Job Scheduling it becomes a challenging task to address them and solve them using necessary job scheduling algorithms.

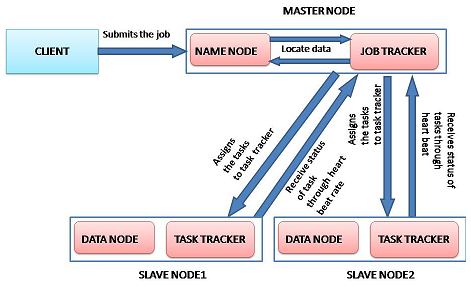
# **II. Hadoop MapReduce: architecture, working Of HAdoop,and issues in Hadoop**

Hadoop basically has two main components which form the core: Hadoop Distributed File system (HDFS) and MapReduce. HDFS [11] is Hadoop distributed file system which is basically used for storing huge files which are normally scattered on various HDFS nodes. Now the data which arrive are normally divided. These data are divided are divided into blocks of 64 MB or 128 MB and are replicated on different machines. Normally three copies of each block on different machine[11]. Now the data is divided into two types:- Metadata and application data. In HDFS there will be a dedicated server known as Namenode which handles metadata and there is a separate node known as Datanode on which application data is maintained.

## Architecture of MapReduce

MapReduce architecture is a very flexible architecture. This architecture can be modified by user with the help of two important functions: Map and Reduce. HDFS is an important component. As mentioned earlier data is divided by HDFS into fixed size blocks and the data is fed into the mapper function to perform parallel map tasks. A key-value pair is generated after the map task which is an intermediate result which in turn will be fed to the Reduce task to obtain the final result. The job of MapReduce is to process the job residing on HDFS. Replication is one of the important feature which is provided by HDFS and is used by MapReduce so as to improve the performance of job scheduling process where scheduling is performed on the basis of node locality and rack locality.

MapReduce architecture mainly consists of two nodes:- Master Node and Slave node. Master node has Job tracker associated to it and Slave node has task tracker associated to it. The role of Job tracker is to manage the slave nodes, known as the TaskTrackers.Apart from this there are two other nodes namely:- Namenode and Data node. Namenode is to maintain and manage namespace which is unique for different types of data. Data node is the one which is used to store HDFS data in the local file system. Based on the different instructions obtained from the Namenode the Data node acts accordingly and stores or retrieves data and passes the necessary information back to the Namenode after performing the necessary action. Following is the architecture of MapReduce which gives a visual description of how the processing takes place and how the different nodes involved in MapReduce perform a specific task.



**Figure:-1** **MapReduce architecture [35]**

## Working of MapReduce

In MapReduce architecture a job consists of various input datasets, a MapReduce program pertaining to it and some configuration properties which are necessary to run the job with a particular scheduler[28]. There are two major tasks in MapReduce namely:- Map task and Reduce task. The job which arrives from the user side is divided to perform Map and Reduce tasks and this is done by Job Tracker. After dividing the tasks are then assigned to the TaskTracker to perform the necessary tasks. Now it’s very important to monitor the tasks by keeping track of the various tasks assigned by communicating with the Task Tracker and is done by Job Tracker. The communication is done with the help of Heartbeat messages. These messages pass the information to Job Tracker whether the task is still performed or not. At the end of the job, JobTracker will inform the status of job to the user. With the help of fixed Map and Reduce slots we can able to determine the number of Map and Reduce tasks that a TaskTracker can able to run at a particular time.

## Issues in MapReduce

A good and effective scheduling policy is very much necessary to manage multiple Map and Reduce tasks in a dynamic environment. With the help of an effective scheduling policy the resources are well utilized and there is no wastage of resources. There are various issues such as Locality, synchronization and Fairness which play a major role in affecting the performance of scheduling policies in a negative way. The issues are as described below:-

### Locality

The distance between input data node and the node on which task is performed is known as Locality[14]. Two types of locality can be found namely:- node locality and rack locality. A rack basically a metal frame, is used to hold various hardware devices such as HDD’s, servers and other different types of electronic equipments. It is always better to assign tasks to the node which consists of input data so that the network cost is reduced and communication can also take place at a very low cost.

### Synchronization

This is also an important issue in scheduling in MapReduce. It is a process where the intermediate outputs obtained from the Map tasks are provided as an input to the Reduce tasks[9]. Reduce tasks are completely dependent on Map tasks. Once the Map tasks get completed only then the Reduce tasks start. It has a bit of a negative impact because since it is dependent the performance of the entire process is affected if any one of the nodes in the cluster slows down. A degradation in the cluster performance can be observed because of this.

### Fairness

This issue basically describes how fairly the resources are being divided amongst the users[10].Sometimes if a heavy job is scheduled to run then there is a possibility that it may take up the entire available resources and the other jobs may be left with no resources for their computation. Because of this it is necessary to divide the resources fairly so that the tasks are performed in a short time. Even if a MapReduce job has an equal share of nodes then also it will result in degradation of throughput and response time; two important factors in Job Scheduling.

# **III. Job Schedulers in Hadoop**

This section provides classification of Hadoop schedulers and discusses various Hadoop schedulers.

## Classification of Hadoop Schedulers

Time, priority, strategy, environment and resource awareness are few of the important parameters on which schedulers are classified. On the basis of resource availability, scheduling strategy and time the schedulers are classified as below:-

### Scheduling strategy:- (Static/Dynamic)

The process where the jobs are allocated to processors even before the execution of the program is known as Static Scheduling[14]. It is only at the compile time that we can get to know the resources involved in job scheduling and its execution time. This type of scheduling aims at reducing the overall execution time of programs.

The process where the jobs are allocated at the time of execution is known as Dynamic scheduling [14]. In this type of scheduling we have prior information about the resources which are needed by a job. It is normally unknown about the environment where the job will be executed. The decision about a job running in a particular environment is made only when it starts its execution in a dynamic environment.

### Resources availability

This type of scheduling is done based on the resource requirement of a job. This scheduling mainly focuses on improving the proper resource utilization and performance of the running job. The common resources that are taken into consideration are:- Time taken by the CPU, storage space of disk, memory, etc.

### Time (Deadline, Delay)

Time constraint scheduling concentrates on scheduling the jobs on the basis of a deadline, i.e. a specific time is fixed for the job to complete.This deadline can be specified by the user and the user can check whether .

## Description of Capacity Scheduler

As Hadoop jobs have to share the cluster resources, a scheduling policy is used to decide when and where (on which machine) a job is to be executed. The objectives of scheduling are to minimize the completion time, maximize throughput, minimize overhead, and balance available resources of a parallel application by properly allocating the jobs to the processors. A detailed description of Capacity Scheduler is as follows:

### Capacity Scheduler

Capacity scheduler [8] developed by Yahoo.is designed keeping multiple organizations in mind which share a large cluster. Because of multiple users it has a provision where the users can access the resources and provides minimum capacity to users. Keeping in mind the dynamic environment many number of queues will be created with appropriate number of Map and Reduce slots. Appropriate amount of capacity is assigned to each queue. Two queues are basically present in Capacity Scheduler namely:- Ready Queue and Waiting Queue. The jobs in the waiting queue are prioritized and priority scheduling is not followed in the ready queue. The resources in the ready queue are equally shared among the jobs.The entire cluster capacity will be equal to the sum of capacities of each queue. The unused capacity is sometimes allotted to the queues in work so as to continue the process. Based on the type of computation the cluster is partitioned keeping in mind the availability of resources in Hadoop.

# **IV. Applying Hadoop MapReduce**

This section discusses about the proposed change in the existing logic used to run jobs in Capacity Scheduler.

## Proposed change in Capacity Scheduler:-

Hadoop supports pluggable schedulers which adds the flexibility of modifying the inbuilt Capacity scheduler to optimize job scheduling performance in a dynamic environment. To improve the utilization of resources and performance of the Hadoop cluster we need to assign the tasks appropriately and efficiently. In the existing system it is mentioned that the jobs in the ready queue will be able to accept the jobs from the waiting queue only when the ready queue is totally empty. This may affect the performance and the jobs may not be able to complete within the expected time specified by the user. To solve this issue a new method is proposed in this paper. In the proposed change in the existing logic the ready queue will be able to accomodate a new job from the waiting queue as soon as the job in the ready queue becomes one less than the maximum number of jobs that it can accommodate and the resources will be divided among the jobs as before. This will help to improve the performance in a dynamic environment and the jobs in the waiting queue won’t have to wait for the all the jobs in the ready queue to get completed. This proposed change will also help the jobs to get completed faster than before since the waiting time is reduce to a large extent.

# **V. Conclusion**

One of the most important and key factor in achieving high performance in Hadoop cluster is Scheduling of Jobs in a dynamic environment. Keeping in mind the various issues in job scheduling such as locality, synchronization, and fairness a new logic for Capacity Scheduler is proposed. Capacity Scheduler was mostly used for short jobs, and most importantly was used to remove the issue of fairness. This paper proposes a method to improve the existing Capacity Scheduler. Since the existing Capacity Scheduler was only suitable for short jobs the proposed method can be very much useful when considering a dynamic scenario which helps in reducing the time taken to complete the jobs in the queue. Apart from that in this paper we have also discussed about MapReduce, the classification of Schedulers and concentrated purely on Capacity Scheduler.

# **References**

1. S. Suthaharan, “Big data classification: Problems and challenges in network intrusion prediction with machine learning,” ACM SIGMETRICS Performance Evaluation Review,  vol. 41, no. 4, pp.70-73, 2014.
2. M. Kaur and Shilpa, “Big Data Visualization tool with Advancement of Challenges,” Int. J. of Advanced Research in Comput. Sci. and Software Eng., vol. 4, 2014.
3. M. Assunção, R. Calheiros, S. Bianchi, M. Netto, and R. Buyya, “Big data computing and clouds: Trends and future directions,” J. of Parallel and Distributed Computing, 2014.
4. G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, “Distributed systems: concepts and design,” Massachusetts: Addison Wesley, 2005.
5. F. Dong and S. G. Akl, “ Scheduling algorithms for grid computing: State of the art and open problems,” Queen’s Univ. Kingston, Ontorio, Tech. Rep., Jan. 2006, vol. 105.
6. S. Bardhan and D. A. Menasce, “The Anatomy of MapReduce Jobs, Scheduling, And Performance Challenges,” Conf. of the Comput. Meseaurement Group, San Diego,CA , 2013.
7. Apache Hadoop. (2014, September 10) [online]. Available: <http://hadoop.apache.org>.
8. J. Chen, D. Wang, and W. Zhao, “A Task Scheduling Algorithm for Hadoop Platform,” J. of Comput., vol. 8, no. 4, pp. 929-936, 2013.
9. Cristian A Gonzalez. (2014, September 24 ) [online]. Available: <http://www.hadooper.blogspot.in>.
10. B. T. Rao and L. S. S. Reddy, “Survey on improved scheduling in hadoop mapreduce in cloud environments,” arXiv preprint arXiv:1207.0780, 2012.
11. D. Borthakur, “The hadoop distributed file system: Architecture and design,” Hadoop Project Website, vol. 11, no. 2007, pp. 21, 2007.
12. A. Rasooli and D. G. Down, “A hybrid scheduling approach for scalable heterogeneous hadoop systems,” High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion:. IEEE, pp. 1284-1291, 2012.
13. G. Coulouris, J. Dollimore, T. Kindberg, and G. Blair, “Distributed systems: concepts and design,” Massachusetts: Addison Wesley, 2005.
14. F. Dong and S. G. Akl, “ Scheduling algorithms for grid computing: State of the art and open problems,” Queen’s Univ. Kingston, Ontorio, Tech. Rep., Jan. 2006, vol. 105.
15. V. K. Dabhi and H. B. Prajapati, “Soft computing based intelligent grid architecture,” In Comput. and Commun. Eng., ICCCE. Int. Conf. on IEEE, pp. 574-577, 2008.
16. T. L. Casavant and J. G. Kuhl, “A taxonomy of scheduling in general-purpose distributed computing systems.” Software Eng., IEEE Trans. on, vol. 14, no. 2, pp. 141-154, 1988.
17. R. I. Davis and A. Burns. “A survey of hard real-time scheduling for multiprocessor systems,” ACM Computing Surveys (CSUR), vol. 43, no. 4, p. 35, 2011.
18. F. Dong and S. G. Akl, “ Scheduling algorithms for grid computing: State of the art and open problems,” Queen’s Univ. Kingston, Ontorio, Tech. Rep., Jan. 2006, vol. 105.
19. H. B. Prajapati and V. A. Shah. "Scheduling in Grid Computing Environment," In  Advanced Computing & Commun. Technologies (ACCT), 2014 4th Int. Conf., IEEE, pp. 315-324, 2014.
20. A. Abraham, R. Buyya, and B. Nath, “Nature’s heuristics for scheduling jobs on computational grids,” In 8th IEEE Int. conf. on advanced computing and commun. (ADCOM 2000), pp. 45-52, 2000.
21. V. K. Dabhi and H. B. Prajapati, “Soft computing based intelligent grid architecture,” In Comput. and Commun. Eng., ICCCE. Int. Conf. on IEEE, pp. 574-577, 2008.
22. H. B. Prajapati and V. A. Shah, “Bandwidth-Aware Scheduling of Workflow Application on Multiple Grid Sites,” J. of Comput. Networks and Commun., 2014.
23. A. Verma, L. Cherkasova, and R. H. Campbell, “ARIA: automatic resource inference and allocation for mapreduce environments,” In Proc. 8th ACM Int. conf. on Autonomic computing ACM, pp. 235-244, 2011.
24. S. Bardhan and D. A. Menasce, “Queuing Network Models to Predict the Completion Time of the Map Phase of MapReduce Jobs,” Proc. of the Comput. Measurement Group Int. Conf., 2012.
25. D. Yoo and K. M. Sim, “A comparative review of job scheduling for MapReduce,” Cloud Computing and Intell. Syst. (CCIS), IEEE Int. Conf. on. IEEE, 2011.
26. S. Sakr, A. Liu, and A. G. Fayoumi, “The family of MapReduce and large-scale data processing systems,” ACM Computing Surveys (CSUR), vol. 46, no. 1, p. 11, 2013.
27. C. Doulkeridi and K. Nørvåg. “A survey of large-scale analytical query processing in MapReduce,” The VLDB J., vol. 23, no. 3, pp. 355-380, 2014.
28. E. Dumbill, “Planning for big data,” O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2012.
29. M. Zaharia, “Hadoop fair scheduler design document,” 2009.
30. C. He, Y. Lu, and D. Swanson, “Matchmaking: A new mapreduce scheduling technique,” Cloud Computing Technology and Sci (CloudCom), 2011 IEEE 3rd Int. Conf. on IEEE, pp. 40-47, 2011.
31. Y. Bu, B. Howe, M. Balazinska, and M. D. Ernst, “HaLoop: Efficient iterative data processing on large clusters,” Proc. VLDB Endowment, vol. 3, no. 1-2, pp. 285-296, 2010.
32. Y. Zhang, Q. Gao, L. Gao, and C. Wang, “imapreduce: A distributed computing framework for iterative computation,”  J. of Grid Computing, vol. 10, no. 1, pp. 47-68, 2012.
33. Hadoop online prototype. (2014, September 28) [online]. Available: https://code.google.com/p/hop/.
34. J. Ekanayake et al., “Twister: a runtime for iterative mapreduce,” Proc. 19th ACM Int. Symp. on High Performance Distributed Computing. ACM, pp. 810-818, 2010.
35. T. Aarnio, “Parallel data processing with MapReduce,” TKK T-110.5190, Seminar on Internetworking., 2009.
36. E. Jahani, M. J. Cafarella, and C.Re, “Automatic optimization for MapReduce programs,” Proc. VLDB Endowment, vol. 4, no. 6, pp. 385-396, 2011.
37. H. Lim, H. Herodotou, and S. Babu, “Stubby: A transformation-based optimizer for mapreduce workflows,” Proc. VLDB Endowment, vol. 5, no. 11, pp. 1196-1207, 2012.
38. H. Herodotou, F. Dong, and S. Babu, “Mapreduce programming and cost-based optimization? crossing this chasm with starfish,” Proc. VLDB Endowment, vol. 4, no. 12, pp. 1446-1449, 2011.
39. H. Herodotou and S. Babu, "Profiling, what-if analysis, and cost-based optimization of MapReduce programs," Proc. VLDB Endowment, vol. 4, no. 11, pp. 1111-1122, 2011.
40. J. Dittrich, J. A. Quiane-Ruiz, A. Zindal, Y. Kargin, V. Setty, and J. Schad, “Hadoop++: Making a yellow elephant run like a cheetah (without it even noticing),” Proc. VLDB Endowment, vol. 3, no. 1-2, pp. 515-529, 2010.
41. Y. Lin, D. Agarwal, C. Chen, B. C. Ooi, and S. Wu, “Llama: leveraging columnar storage for scalable join processing in the MapReduce framework,” Proc. 2011 ACM SIGMOD Int. Conf. on Manage. of data. ACM, pp. 961-972, 2011.
42. Y. Kwon, M. Balazinska, B. Howe, and J. Rolia, “Skewtune: mitigating skew in mapreduce applications,” Proc. 2012 ACM SIGMOD Int. Conf. on Manage. of Data. ACM, pp. 25-36, 2012.
43. M. Yong, N. Garegrat, and S. Mohan, “Towards a resource aware scheduler in hadoop,” Proc. ICWS., pp. 102-109, 2009.
44. Hadoop Wiki. (2014, October 3) [online]. Available: http://wiki.apache.org/hadoop/GangliaMetrics.
45. Nagios Exchange (2014, October 3) [online]. Available: http://exchange.nagios.org/.
46. Apache Ambari. (2014, October 3) [online]. Available: http://ambari.apache.org/.
47. T. White, “Hadoop: The definitive guide,” 2nd ed., O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2012.
48. E. A. Chinnu and J. S. Manjaly, “A Relative Study on Task Scheduling in Hadoop MapReduce,” ijrcsse, vol. 3, Issue-5, 2013.
49. “Hadoop Performance Tunning”, Impetus Technologies Inc., October, 2009.
50. K. Lee, Y. Lee, H. Choi, Y. D. Chung, and B. Moon, “Parallel data processing with MapReduce: a survey,” AcM sIGMoD Record, vol. 40, no. 4, pp. 11-20, 2012.