ISSN No. 0976-5697

Volume 8, No. 3, March – April 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

Comparative study of Job Schedulers in Hadoop Environment

Arpitha HV and Shoney Sebastian
Department of Computer Science
Christ University, Bengaluru, India

***Abstract*:** Hadoop is a structure for BigData handling in distributed applications. Hadoop bunch is worked for running information intensive distributed applications. Hadoop distributed file system is the essential stockpiling territory for BigData. MapReduce is a model to total undertakings of a job. Task assignment is conceivable by schedulers. Schedulers ensure the reasonable assignment of assets among clients. At the point when a client presents a job, it will move to a job queue. From the job queue, job will be divided into tasks and distributed to various nodes. By the correct assignment of tasks, job finish time will decrease. This can guarantee better execution of the job. This paper gives the comparison of different Hadoop Job Schedulers.

***Keywords*:** Hadoop, HDFS, MapReduce, Scheduling, FIFO Scheduling, Fair Scheduling, Capacity Scheduling.

I. INTRODUCTION

Hadoop [1,2] is structure for distributed applications. Doug Cutting is motivated by Google MapReduce [3] and he presented Hadoop MapReduce for distributed applications. Hadoop primarily manages BigData preparing on distributed environment. Hadoop distributed file system (HDFS) is the storage zone in this system. MapReduce is the method for BigData preparing and analyzing in parallel. The main target of Hadoop framework is to hide the points of parallel processing that includes data distribution to handling nodes, restarting fizzled subtasks and consolidation of results after the computation. This open source system permits software engineers to execute parallel processing programs that concentrate on their computational issues. There are two major components of Hadoop (1) Hadoop distributed file system (HDFS): which is utilized for storing large amount of datasets with high degree of throughput. These huge datasets are stored on number of groups. (2) Map Reduce: It is a software framework for preparing enormous datasets with the clusters of commodity servers through parallelization.

Hadoop Distributed File System (HDFS) [4,5] stores vast documents over numerous machines. It accomplishes dependability by replicating the data over numerous servers. Similarly to GFS, various copies of data are put away on different compute nodes to give reliable and rapid computations. Data is also given over HTTP, permitting access to all substance from a web program or different types of users. HDFS has master/slave architecture. HDFS consists of a single NameNode and multiple DataNodes in a cluster. NameNode is in charge for mapping of data blocks to DataNodes and for managing file framework operations like opening, shutting and renaming documents and registries. Upon the instructions of NameNode, DataNodes perform block creation, deletion and replication of data blocks. The NameNode likewise keeps up the file framework namespace which records the creation, cancellation and change of records by the clients. NameNode decides about replication of data blocks. In a normal HDFS, block size is 64MB and replication factor is 3.

MapReduce [6,7] is one of the parallel data processing paradigm designed for vast scale information preparing on group based computing structures. It was originally proposed by Google to deal with large scale web search applications. This approach has been proved to be an successful programming approach for developing machine learning, data mining, and search applications in data centers. Its advantage is that it permits software engineers to extract from the issues of scheduling, parallelization, partitioning, replication and concentrate on developing their applications. Hadoop MapReduce programming model consists of data preparing functions: Map and Reduce. Parallel Map tasks are keep running on data which is divided into fixed sized blocks and create intermediate output as a collection of <key, value> sets. These sets are rearranged crosswise over various decrease tasks based on <key, value> sets. Each Reduce task accepts just a single key at once and process data for that key and yields the outcomes as <key, value> sets. The Hadoop Map Reduce design consists of one Job Tracker (Master) and numerous Task Trackers [8] (Workers). The Job Tracker gets job submitted from client, breaks it down into map and reduce tasks, assigns the tasks to Task Trackers, screens the advance of the Task Trackers, lastly when every one of the tasks are finished, reports the client about the job completion. Every Task Tracker has a fixed number of map and reduce task slots that decide what number of map and reduce tasks it can keep running at once. HDFS supports reliability and fault tolerance of MapReduce computation by storing and replicating the inputs and outputs of a Hadoop work. Since Hadoop jobs need to share the bunch resources, a planning strategy is utilized to decide when a job can execute its tasks. Earlier Hadoop had an extremely basic scheduling algorithm works on First-in First-out (FIFO) basis for scheduling client's jobs by default. Later huge measure of research occurred in growing more successful and environment-specific schedulers. Every one of those schedulers will be explained in the following area.

Job Schedulers in Hadoop

In Hadoop, schedulers are playing a fundamental part in job task. FIFO Schedulers, Fair Schedulers and Capacity Schedulers are the default schedulers in Hadoop and later, many improved schedulers were introduced. In a distributed environment, task scheduling is more effective because

completion of job execution turns out to be speedier and it brings about enhanced execution.

Task assigning is one of the important procedures in Hadoop. Schedulers are in charge of doing task assignment. Two types of hubs called JobTracker and TaskTracker are participating in the job execution handle. JobTracker goes about as a Coordinator of all jobs executing in the machine while the TaskTracker executes the assignments and sends a continuous answer to the JobTracker.

II. LITERATURE REVIEW

B P Andrews and A Binu [9] talks about FIFO Scheduler. For job scheduling Hadoop gives default FIFO scheduler where jobs are scheduled in FIFO order. But this scheduler might not be a good decision for a few jobs. So in such situations one should choose an alternate scheduler.

S Santhosh and H Kumar G [10,11] talks about improved Fair Scheduling Algorithm in Hadoop Environment. Job scheduling is an important procedure in Hadoop Map Reduce. Hadoop comes with three types of schedulers namely FIFO, Fair and Capacity Scheduler. Here, fair sharing of resources is possible. Main advantage of the scheduler is that whenever slot becomes free, shorter jobs can be assigned. Unlike to FIFO scheduler, the smaller jobs need not hold up with a specific end goal to finish a big job. Main disadvantage is that tasks will be distributed to every one of the slots in the cluster with maximum slot capacity. In fair scheduling, pool of jobs will be there.

S Thakur et al., [12,13] talks about Capacity Scheduling in Hadoop. This paper proposed the improved capacity scheduler to improve the current scheduler issues that help the scheduler to execute the task in less time. They presented pipeline and queue management in their proposed work for improving the performance of hadoop.

M. Zaharia et al., [14] talks about Delay Scheduling, there is a clash between fairness in scheduling and data locality. They illustrate this issue through their experience planning a fair scheduler for a 600-node Hadoop cluster at Facebook. To address the clash between locality and fairness, they propose a simple algorithm called Delay Scheduling.

T Sandholm and K Lai [15] talks about Dynamic Priority Scheduler. It permits clients to control their allocated capacity by adjusting their spending over time. This basic mechanism permits the scheduler to make more effective decision about which jobs and users to prioritize and gives clients the tool to optimize and customize their allocation to fit the importance and requirements of their jobs. Moreover, it gives clients the motivation to scale back their jobs when request is high, since the cost of running on a slot is then also more expensive. They envision their scheduler to be used by deadline or budget optimized agents on behalf of clients. They describe the design and implementation of the Dynamic Priority scheduler and experimental results.

A Raut et al., [16] talks about Deadline Constraint Scheduler in Hadoop. Given a question q that means a MapReduce job J and needs to process data of size σ be completed within a deadline D, when run on Hadoop Yarn architecture having N heterogeneous nodes, how successfully can the jobs be scheduled is explained in this paper. K. Kc and K. Anyanwu [17] dicusses that Deadline Constraint Scheduler enhances framework use managing with deadline requirement and data preparing. It is accomplished by cost model for job execution and Hadoop scheduler with limitation. Cost model with job execution considers different parameters like MapReduce task with runtime, the input size of data and data distribution. A Hadoop scheduler with client constraints has deadline as part of the input; when the job is submitted for testing, it checks whether the job can be done inside the time determined by the deadline or not.

J S Manjaly and C Edwin A [18] dicusses that in learning Scheduler, jobs are classified as good or bad. Design classifier arranges the jobs. As per the resource usage, good jobs will be considering for further handling. Good job does not make any over-burden to the TaskTrackers. Bad jobs will be rejected. The scheduler considers CPU use, Memory usage, IO use and Network use. In the event that more than one good job land in the job line, job will be chosen by the more expected utility capacity. Task assignment is like default schedulers.

S Kalra and A lamba [19] talks about Resource aware scheduler. This Scheduler in Hadoop has turned out to be one of the Research Challenges in Cloud Computing. Scheduling in Hadoop is centralized, and worker initiated. Scheduling choices are taken by a master hub, called the Job Tracker. The Job Tracker keeps up a queue of currently running jobs, conditions of Task Trackers in a group, and list of tasks allocated to every Task Tracker. Every Task Tracker hub is as of now arranged with a most extreme number of accessible computation slots. Every Task Tracker hub screens assets, for example, CPU usage, disk channel IO in bytes/s, and the quantity of page faults per unit time for the memory subsystem. Two of its approaches are:

- Dynamic Free Slot Advertisement
- Free Slot Priorities/Filtering

M Zaharia et al., [20] talks about Longest Approximate Time to end scheduler. At some point in Hadoop, task will be finished gradually, because of substantial load on the hub, some failures might be there or slow background processes. The scheduler will discover the slow running task to dispatch another task as a backup task that is referred as theoretical execution of tasks. To choose theoretical tasks, scheduler monitors tasks advance utilizing an advance score between 0 and 1. On the off chance that the foundation work finishes quicker, the job execution is moved forward. Scientist have proposed another technique for theoretical execution called Longest Approximate Time to End (LATE) algorithm that uses an alternate metric to schedule tasks for speculative execution. This strategy would be ideal if hubs kept running at steady speeds and if there was no cost to launching a theoretical task.

A Rasoolia and D G Down [21] discusses about COSHH, which is designed and executed for Hadoop, it considers heterogeneity at both application and group levels. The principle approach in COSHH scheduling framework is to utilize framework data to settle on better scheduling choices, which prompts to enhancing the execution. COSHH consists of two principle forms, where every procedure is activated by getting one of these messages. After accepting another job, the scheduler performs the queuing process to store the incoming job in a appropriate queue. After accepting a pulse message, the scheduler triggers the

directing procedure to allocate a vocation to the present free asset. COSHH is proposed to enhance the mean completion time of jobs.

P Kondikoppa et al., [22] discusses about Network Aware Scheduler in Hadoop. Data locality is the main issue for undertaking relegating in numerous schedulers. Network Aware scheduler gives a administrator control script to discover the area of the hub that has the information required for the task. Up to a limit, data locality issue can be expelled in this scheduler. Here, FIFO and Fair scheduler is stretched out with network awareness. At the point when an undertaking is alloted to the TaskTracker, the information required for executing the task will be checked. If it is available, then the task is assigned to the space. Generally the scheduling of the task is deferred for particular delay.

III. PROPOSED WORK

In this paper our work is to compare different Job Schedulers in Hadoop i.e. FIFO Scheduling, Fair Scheduling, Capacity Scheduling, Delay Scheduling, Dynamic Priority Scheduler and Deadline Constraint Scheduler. We have compared different properties like Job allocation, priority in job queue, Fairness/Fair sharing resources, locality problem, mode, taxonomy, Sticky Slots, advantages, disadvantages and environment of different Hadoop Job Schedulers.

We have collected some twenty research papers of comparison on different Job Schedulers in Hadoop. Out of these comparisons we combined many common parameters in Job schedulers and we done three tables giving comparison on different Job Schedulers in Hadoop. This paper helps to identify the different properties of Job Scheduers and also the advantages and disadvantages of different Job schedulers in Hadoop.

IV. EXPERIMENTAL RESULT.

Table 1: Comparative study of Different Job Schedulers in Hadoop.

Name of the Scheduler	Job allocation	Priority in job queue	Fairness/ Fair sharing of Resources	Locality problem	Mode	Taxonomy	Sticky Slots
FIFO Scheduler	Static	No	No	Yes	Non Preemptive	Non adaptive	NA
Fair Scheduler	Static	Yes	Fair share of the cluster capacity over time	YES for small jobs	Preemptive	adaptive	Yes
Capacity Scheduler	Static	No	Yes	Yes	Non Preemptive when job fail	adaptive	NA
Delay Scheduler	Static	Yes	Less Fairness than Fair Scheduler	Improved compared to Fair scheduler	Preemptive	adaptive	NA
Dynamic Priority Scheduler	Dynamic	Yes	Yes	No	Preemptive	adaptive	NA
Deadline Constraint Scheduler	Dynamic	Yes	Yes	Yes for small jobs	Preemptive	adaptive	NA
LATE Scheduler	Static	Yes	Yes	Yes	Preemptive	adaptive	NA
COSHH	Dynamic	Yes	Yes	No	Preemptive	Adaptive	Yes

Table 1 give the comparative study of different schedulers. It is visible that Capacity scheduler and Fairness scheduler are used for resolving fairness issues in short jobs and production jobs. The data locality issue which cannot be solved by FIFO Scheduler, Fair Scheduler and Capacity Scheduler. Hadoop uses static job allocation policy and the number of Map/Reduce slots is fixed. Hadoop cluster with resource aware cluster provides cluster utilization, minimizing resource consumption. Job scheduling algorithm is an important research direction in Big Data processing.

Table 2: Environment of different Job Schedulers in Hadoop

Scheduling Algorithm	Environment	
	Homogeneous	Heterogeneous
FIFO	✓	✗
Fair Scheduling	✓	✗
Capacity Scheduling	✓	✗
Delay Scheduling	✓	✗
Dynamic Priority Scheduler	✓	✗
Deadline Constraint Scheduler	✓	✓
LATE Scheduler	✓	✓
COSHH Scheduler	✓	✗

Table 3: Comparison on advantages and disadvantages of Job Schedulers in Hadoop

Scheduling Algorithms	Advantages	Disadvantages
FIFO Scheduling	Easy to understand and equally easy to program	It pays no attention to processing time or prioritizes the processes
Fair Scheduling	1. Less complex. 2. Works well when both small and large clusters. 3. It can provide fast response times for small jobs mixed with larger jobs.	Does not consider the job weight of each node.
Capacitive Scheduling	Ensure guaranteed access with the potential to reuse unused capacity and prioritize jobs within queues over large cluster.	The most complex among three schedulers
Delay Scheduling	Simplicity of scheduling	No particular
Dynamic Priority Scheduler	can be easily configured	If the system eventually crashes then all unfinished low priority processes gets lost.
Deadline Constraint Scheduler	Helps in Optimization of Hadoop implementation	Nodes should be uniform in nature which incurs cost

V. CONCLUSION

This paper gives an overall idea about different Job Schedulers in Hadoop MapReduce. It compares the properties of FIFO scheduling, Fair scheduling, Capacity scheduling, Delay Scheduling, Dynamic Priority Scheduler, Deadline Constraint Scheduler, LATE Scheduler and COSHH Scheduler. We have compared different properties like Job allocation, priority in job queue, Fairness/Fair sharing resources, locality problem, mode, taxonomy, Sticky Slots, advantages, disadvantages and environment of different Hadoop Job Schedulers. This paper helps to identify the different properties of Job Scheduers and also the advantages and disadvantages of different Job schedulers in Hadoop.

VI. REFERENCES

[1] H S Bhosale, D P Gadekar, "A Review Paper on Big Data and Hadoop", International Journal of Scientific and Research Publications, vol. 4, Issue 10, 2014.

[2] H S Bhosale, D P Gadekar, "Big Data Processing Using Hadoop: Survey on Scheduling", International Journal of Science and Research (IJSR), Vol. 3 Issue 10, 2014.

[3] S Ghemawat, "The Google file system", In 19th Symposium on Operating Systems Principles, Lake George, New York, 2003.

[4] K Shvachko et al., "The Hadoop Distributed File System", Mass Storage Systems and Technologies, Pp 1-10, 2010.
[5] Hadoop Distributed File System: http://hadoop.apache.org/docs/r1.2.1/hdfs_design.html
[6] J Dean and S Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters", Operating System Design and Implementation, 2014.
[7] T Condie et al., "MapReduce Online", Networked Systems design and implementation, Pp 21-21, 2010.
[8] http://hadoopinrealworld.com/jobtracker-and-tasktracker/
[9] B P Andrews and A Binu, "Survey on Job Schedulers in Hadoop Cluster", IOSR Journal of Computer Engineering, vol. 15, Issue 1, Pp 46-50, 2013.
[10] S Santhosh and H Kumar G, "IMPROVED FAIR SCHEDULING ALGORITHM FOR TASKTRACKER IN HADOOP MAP-REDUCE", International Journal of Advanced Technology in Engineering and Science, vol. 03, Issue 01, 2015.
[11] V P Narkhede, S T Khandare, "Fair Scheduling Algorithm with Dynamic Load Balancing Using In Grid", Research Inventy: International Journal Of Engineering And Science Vol.2, Issue 10, Pp 53-57, 2013
[12] S Thakur et al., "Dynamic Capacity Scheduling in Hadoop", International Journal of Computer Applications, vol. 125, Issue 15, 2015.
[13]Hadoopjobscheduling:http://blog.cloudera.com/blog/2008/11/job-scheduling-in-hadoop/
[14] M Zaharia et al., "Delay Scheduling: A Simple Technique for Achieving Locality and Fairness in Cluster Scheduling", EuroSys '10, Pp 265-278, 2010.
[15] T Sandholm and K Lai, "Dynamic Proportional Share Scheduling in Hadoop", Job Scheduling strategies for parallel processing, Pp 110-131, 2010.
[16] A Raut et al., "Deadline aware scheduler for Hadoop Yarn", IEEE International Parallel and Distributed Processing Symposium, Pp 956-965, 2012.
[17] K Kc and K Anyanwu, "Scheduling Hadoop Jobs to Meet Deadlines", Cloud Computing Technology and Science, Pp 388-392, 2010.
[18] J S Manjaly and C Edwin A, "A Relative Study on Task Schedulers in Hadoop MapReduce", International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, Issue 5, 2013.
[19] S Kalra and A lamba, "A Review on HADOOP MAPREDUCE-A Job Aware Scheduling Technology", International Journal of Computational Engineering Research (IJCER), Vol. 04, Issue 5, 2014.
[20] M Zaharia et al., "Improving MapReduce Performance in Heterogeneous Environments", Operating systems design and implementation, Pp 29-42, 2008 .
[21] A Rasoolia and D G Down, "COSHH: A Classification and Optimization based Scheduler for Heterogeneous Hadoop Systems", Future Generation Computer Systems, 2014.
[22] P Kondikoppa et al., "Network-Aware Scheduling of MapReduce Framework on Distributed Clusters over High Speed Networks", Cloud services, federation, and the 8th open cirrus summit, Pp 39-44, 2012.