Grenze Int. J. of Engineering and Technology, Vol. 1, No. 1, January 2015

Survey on Hadoop MapReduce Scheduling Algorithms

[1]Pranoti K. Bone and [2]A.M.Wade
[1]PG Student and [2]Asst. Professor
[1-2]Computer Engineering Depatment, Smt. Kashibai Navale College of Engineering, Vadgaon(Bk), Pune, India.
pranotibone7@gmail.com adi.wade@gmail.com

***Abstract*— Mapreduce in hadoop is parallel processing. In modern world data centres operate various MapReduce function in parallel, hence it is essential to furnish an effective scheduling algorithm in order to optimize completion time required for these jobs. Hadoop adapted FIFO scheduling as default scheduling algorithm but maybe this scheduling is not powerful to fulfil the requirements of all jobs. So in this situation one should use alternate scheduling algorithms. This study will incite other experience clients and designers to conceive the details of certain scheduling, enable them to make the best decisions for their certain research interests.**

***Index Terms*— Hadoop, MapReduce, scheduling algorithm.**

I. INTRODUCTION

Many big firms like Amazon, Facebook and Yahoo use Hadoop. Hiding the details of parallel processing, including data distribution to processing nodes is possible just because of Hadoop [1]. There are the two main component of Hadoop - 1.Hadoop Distributed file system (HDFS) 2.Hadoop MapReduce [2].
HDFS is called as block oriented file system shown in figure 1 [3]. Here every individual file is divided into a block of 64MB. Further these blocks are stored within machines having cluster along with data storage capacity. Every individual machine in the cluster is denoted as 'DataNode'. Every file constitutes several blocks which are not stored on the same machine .On block by block basis these target machines holding each block are selected randomly. Hence in order to access certain file the co-operation of multiple machine is required. Because of failure of any node in cluster problem of unavailability arises. HDFS cures this problem by making copies of each block over number of machines, generally it is taken as 3.
NameNode is a single node in HDFS cluster. It not only manages file system namespace but also regulates user access to files. DataNode helps to store data in block format within files. There are two functions are performed by NameNode - 1. To map data block to DataNode 2. To manage file system. Operations like opening, closing, renaming files and directories. In case of failure of NameNode machines, NameNode data must be preserved. Numbers of copies of NameNode information are maintained on different machines. Hence, in the event of crash it can be accessed by other nodes in cluster. These other nodes can be denoted as secondary NameNode.
MapReduce was primarily suggested by Google in order to manage large scale web search applications. MapReduce is signified as an effective programming approach because of advancement of machine learning, data mining and search application in data centres. It includes following two data processing functions - 1)

Grenze ID: 01.GIJET.1.1.24

Map 2) Reduce.

Parallel Map functions are carried on input data which is divided into predetermined sized blocks and generate intermediate output as a cluster of <key, value>pairs. These pairs are intermixed along with various reduce tasks based on <key, value>pairs. Hadoop MapReduce has a master – workers like architecture means it consists of one JobTracker i.e. master and various TaskTracker i.e. workers. The duty of separation from input data and determination of TaskTracker depending on their network space to data source is entrusted upon JobTracker .On the other hand TaskTracker have to submit periodic status report to their master i.e. JobTracker through heartbeat message.

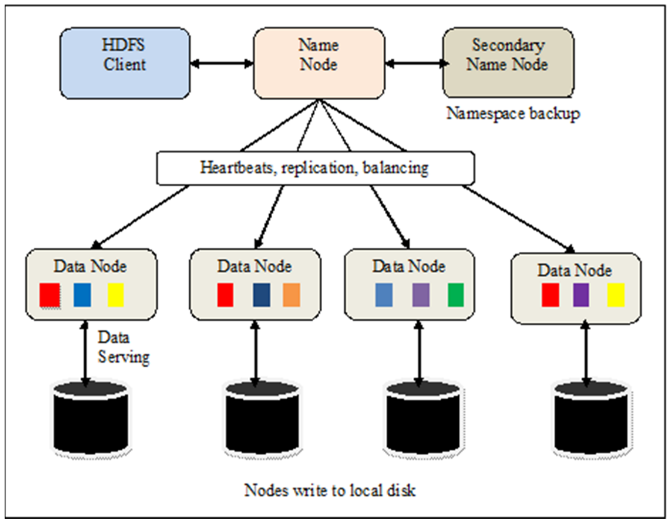

Fig.1. HDFS Architecture

TABLE I. MAPREDUCE I/O

Mapreduce I/O			
Functions	Input	Output	Directions
Map	(K1,V1)	(K2,V2)	The input keys (K1,V1) is mapped to keys k of an intermediate format (K2,V2) collection.
Reduce	(K2,V2)	(K2,V2)	Reduce a group of middle set values associated with K2 to smaller set of values.

In the event of task or worker failures, functions are relaunched on other nodes. The records of heartbeat messages coming from TaskTracker are maintained by JobTracker in order to use them in task assignment. In modern world data centres operate various MapReduce function in parallel, hence it is essential to furnish a effective scheduling algorithm in order to optimize completion time required for these jobs. Nowadays scheduler gives more attention to optimized, little theoretical understanding of scheduling problem subsists in relation to MapReduce. Here we have scrutinized the scheduling algorithm for MapReduce and also compared distinct scheduling algorithm [4][5][6] for MapReduce framework for Hadoop.

II. SCHEDULING IN HADOOP

A. Issues Of Schedulinng In Mapreduce

- *Locality* - Locality is the major issue of map-reduce scheduling. The distance between the input data node and task-assigned node is termed as Locality. Lesser distance leads to lesser data transfer cost. As compared to other scheduling constraints locality is considered as basic approach. Because of limited bisection bandwidth of network locality is considered as very critical issue affecting performance in shared cluster environment. Throughput of task increases due to high locality. Node locality is defined as the processing of a task on a node holding the data. In case of impossibility of achieving node locality, job is executed on the same track called as rack locality. In case of nonfulfillment of locality Data transferring I/O cost can greatly affect the execution because of shared bandwidth of network. In order to save cost of network, jobs pursue the policy of assigning task to the nearer location.

- *Synchronisation* - The process of transmitting the intermediate output of the map function to the reduce operation as input is also taken into account as a component which affect performance is termed as synchronisation. Before initiating the sending of intermediate output mappers have to wait for the completion of all the map processes. As map and reduce phases are interdependent, a single node slows down the whole procedure by making other nodes wait till completion. There are many components which degrades the synchronisation process. For e.g. node failures miss-configuration, heterogeneity of cluster and serious overhead of I/O cost.
- *Fairness* - In many big enterprises like Yahoo, Facebook, Google several map-reduce jobs are executed in the shared data warehouse of respective firm. As map-reduce function possess heavy workload , it may rule the utilization of shared clusters, hence some short computation tasks may not have.

B. Related Work

Hadoop uses various scheduling algorithms for task assignment. Many researchers are focusing on this issue. Various scheduling algorithm are listed in table I.

TABLE II. VARIOUS SCHEDULING ALGORITHMS

Scheduling Algorithm	Description	Advantages	Disadvantages
FIFO Scheduling	First in first out – Oldest job selected first by job tracker.	• The HOD virtual cluster can be utilised in a comparatively self-directing way. • It is also adjustable in that it can dwindle when the workload varies. • When it has no running function, it automatically de allocates node from virtual cluster. • It supplies greater safety, with less sharing of nodes. • Because of lack of dissension within the nodes for multiple clients' job it enhance the performance. • Poor Locality • Poor utility	• Size of the job or priority are not given any importance • As the resources are used by large jobs, small jobs are ignored.
Fair Scheduling	A Process of allocating resources to jobs in such a way that every job will get almost same proportions of resources. When there is any slot vacant the scheduler will assign this slot to the job having huge job deficit.	• Less complex • Works well when both small and large clusters • Furnish fast response time for small tasks mixed with large tasks. • It sets the bounds to the number of associate jobs in every job pull.	• Job size is completely ignored. • Does not examine availability of resources on fine-grained basis.
Capacity Scheduling [8]	The capacity scheduler grants sharing huge cluster by devoting each firm capacity guarantee. In case of vacant slots in certain JobTracker, the scheduler will select a queue, then select a job and at last allocate this slot to a job.	Ensure guaranteed access with the potential to reuse unused capacity and prioritize jobs within queues over large cluster.	• The most complex among three schedulers. • Does not consider resource availability on a fine-grained basis.

Delay scheduling [9][10]	Queue based scheduling relaxing fairness for locality enhancement. Although the first slot we consider giving to a job is unlikely to have data for it, tasks finish so quickly that some slot with data for it will free up in the next few seconds	Simplicity of scheduling	No particular
Dynamic priority scheduling [11]	In order to balance current workload it allows users to enhance or diminish their queue priorities. Supports the capacity distribution dynamically among concurrent users based on priorities of the users. As per priorities of the users. It helps the capacity distribution.	Configured easily.	In the event of crash of system all incomplete low priority processes gets lost.
Deadline constraint scheduling [12]	It concentrates on the deadline constraint of task which denotes the problem of deadline but mainly helps in enhancing system utilization.	Supports optimization of hadoop implementation.	Cost incurred for each node should be uniform.
Longest Approximate Time to End Scheduling [13]	LATE scheduler always speculatively executes the task. If any task works slowly so it is very uncommon to continue with the task processing. Task is progress is very slow due to some reasons like high CPU load on the node, slow background processes etc. All tasks should be finished for completion of the entire job. The scheduler detects a slow running task to launch another equivalent task as a backup which is termed as execution relies implicitly on certain assumptions: a) Uniform Task progress on nodes b) Uniform computation at all nodes.	• Robust to node heterogeneity. • Considers node heterogeneity while determining where to run speculative jobs. • Rather than executing task having slow response time it speculatively executes only the jobs which will enhance job response time	• It does not divide map slow nodes and decrease slow nodes. • The static manner of computing the growth of job results in poor
Self-Adaptive MapReduce Scheduling [14]	The process of SAMR algorithm includes reading the historical information and tuning parameters, finding the slow tasks, finding the slow TaskTracker, launching backup tasks, collecting results and updating the historical information.	Uses historical information to tune weights of map and reduce stages.	It does not consider that the dataset sizes and the job types can also affect the stage weights of map and reduce tasks.
Enhanced Self-Adaptive MapReduce Scheduling [15]	ESAMR scheduling algorithm also considers the fact that slow tasks extends the execution time of the whole job and due to hardware heterogeneity different amounts of time are needed to complete the same task on different nodes. ESAMR records historical information for each node as in case of SAMR and it adopts a k-means clustering algorithm to dynamically tune stage weight parameters and to find slow tasks accurately. ESAMR significantly improves the performance of MapReduce scheduling in terms of estimating task execution time and launching backup tasks	• Can identify slow tasks more accurately. • Improves the performance in terms of estimating task execution time and launching backup tasks.	• Little overhead due to K-means algorithm. • Allows only one speculative copy of a task to run on a node at a time.

III. CONCLUSION

Now a day, the demand of Hadoop is greatly enhancing. Industry possess a large amount of data in large number of data sets hadoop can be implemented. MapReduce is one of the most important parts of hadoop. Here in this paper we have discussed many scheduling algorithms. In the scheduling of hadoop there are several research channels. Future work includes development of scheduling algorithm for load balancing strategy.

ACKNOWLEDGMENT

I would like to thank my respected Guide Prof. A. M. Wade for encouragement and support. I gratefully acknowledge him for inculcating valuable basic knowledge of Big Data Hadoop, MapReduce in me. We are also thankful to the department of Computer Engineering, Smt. Kashibai Navale College of Engineering, Pune for providing us infrastructure facilities necessary for this work and moral support.

REFERENCES

[1] Apache Hadoop. http://hadoop.apache.org.
[2] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. OSDI '04, pages 137–150, 2004.
[3] Hadoop Distributed File System, http://hadoop.apache.org/hdfs.
[4] V. Krishna Reddy, B. Thirumala Rao, LSS Reddy, "Research issues in Cloud Computing", Global Journal Computer Science & Technology Vol. 11, no. 11, June 2011,pp.70-76.
[5] Yang XIA†, Lei WANG1, Qiang ZHAO1, Gongxuan ZHANG2, "Research on Job Scheduling Algorithm in Hadoop", 2011.
[6] Stonebraker, M., "MapReduce and parallel DBMS: friends or foes?", ACM, 2010.
[7] B. Thirumala Rao, N. V. Sridevi, V. Krishna Reddy, LSS.Reddy, "Performance Issues of Heterogeneous Hadoop Clusters in Cloud Computing", Global Journal Computer Science & Technology Vol. 11, no. 8, May 2011,pp.81-87.
[8] Hadoop's Capacity Scheduler :http://hadoop.apache.org/core/docs/current/capacityscheduler.
[9] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma, Khaled Elmeleegy, Scott Shenker, and Ion Stoica. "Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling in EuroSys10", Proceedings of the 5th European conference on Computer systems, pages 265–278, New York, NY, USA, 2010. ACM.
[10] Matei Zaharia, Dhruba Borthakur, Joydeep Sen Sarma,Khaled Elmeleegy, Scott Shenker, Ion Stoica, "Job Scheduling for Multi-User MapReduce Clusters", Electrical Engineering and Computer Sciences, University of California at Berkeley, April 2009.
[11] Thomas Sandholm and Kevin Lai. "Dynamic proportional share scheduling in Hadoop in JSSPP", 15th Workshop on Job Scheduling Strategies for Parallel Processing, April, 2010.
[12] K. Kc and K. Anyanwu, "Scheduling Hadoop Jobs to Meet Deadlines", in Proc. CloudCom, 2010, pp.388- 392.
[13] M.Zaharia, A.Konwinski, A.Joseph, Y.zatz, and I.Stoica. Improving mapreduce performance in heterogeneous environments. In OSDI'08: 8th USENIX Symposium on Operating Systems Design and Implementation, October 2008
[14] Quan Chen; Daqiang Zhang; Minyi Guo; Qianni Deng; Song Guo; , "SAMR: A Self-adaptive MapReduce Scheduling Algorithm in Heterogeneous Environment,"(2010) Computer and Information Technology (CIT), 2010 IEEE 10th International Conference on , vol., no., pp.2736-2743.
[15] Xiaoyu Sun, Chen He and Ying Lu "ESAMR: An Enhanced Self-Adaptive MapReduce Scheduling Algorithm"(2012) IEEE 18th International Conference on Parallel and Distributed Systems.