Volume 6, Issue 10, October 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Adhoc Query Implementation Using Unfair Slots Scheduling in Hadoop

[1]Anu Sharma, [2]Monika Khurana
[1] M.Tech Student, [2]Assistant Professor
[1, 2] Department of Computer Science and Engineering, SDDIET, Barwala, Panchkula, Haryana, India

Abstract- Hadoop is a Java-based computing framework that supports the storing and processing of big data in a distributed computing environment and it is suitable for large volume of data. it use hadoop distributer file system for storing and map reuduce to processing the data. the aim of MapReduce programming model is to parallelize the job execution across multiple nodes for execution. Now a days, all focus of the researchers and companies toward to Hadoop. Because of this, many scheduling algorithm have to be introduced . The existing algorithm use fare share of slots for execution the jobs on single node.There is the problem of congestion. That's why the research work is motivated to assign a set of unfare shares to each pool to balance resources across jobs in pools. the unfare share congestion can be minimized and the performance of hadoop can be increased.

Keywords: Hadoop, Map Reduce, Locality, Synchronization, Fairness ,Scheduling algorithms.

I. INTRODUCTION

Hadoop is an open source,distributed computing framework for big data storage and analytics. This takes care of detecting and handling failures. This ensures availability of data by creating multiple copies of the data in different nodes throughout the cluster. In Hadoop, the code is moved to the location of the data instead of moving the data towards the code. It is a highly reliable and efficient cloud computing platform, which can be deployed in a cloud computing data center. the users of hadoop need not concern on the low-level details of distributed system but focus on their business need when they develop distributed applications. Because of this lots of famous Internet service providers, including Twitter, Yahoo, Baidu, and Alibaba etc have already chosen Hadoop as one of the core components to build their own cloud systems to provide more efficient services.

1.1 How is Hadoop architected?

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, the data divided 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 storeage of multiple copy, data stored on a server that goes offline or dies can be automatically replicated from a known good copy. Hadoop has Hadoop Distributed File System (HDFS) and Hadoop MapReduce: a software framework for distributed processing of data on cluster.

A. HDFS- Distributed file system

Hadoop Distributed File System (HDFS) stores large files across multiple machines. It achieves reliability by replicating the data across the multiple servers. Files are divided into chunks of 64 MB, and are usually appended to or read and only extremely rarely overwritten . Compared with traditional file systems, GFS is designed and optimized to run on data centers to provide extremely high data throughputs, low latency and survive individual server failures. I Multiple replicas of data are stored on multiple compute nodes to provide reliable and rapid computations. Data is provided over HTTP, allowing access to all content from a web browser or other types of clients. HDFS has master/slave architecture.it has namenode and datanode.[2].

As shown in Fig. HDFS consists of a single NameNode and multiple DataNodes in a cluster. NameNode is for mapping of data blocks to DataNodes and for managing file system operations like opening, closing and renaming directories and files. Upon the instructions of NameNode, DataNodes perform block creation, deletion and replication of data blocks. The NameNode also maintains the file system namespace which records the creation, deletion and modification of files by the users. NameNode decides about replication of data blocks. In a typical HDFS, block size is 64MB .[9]

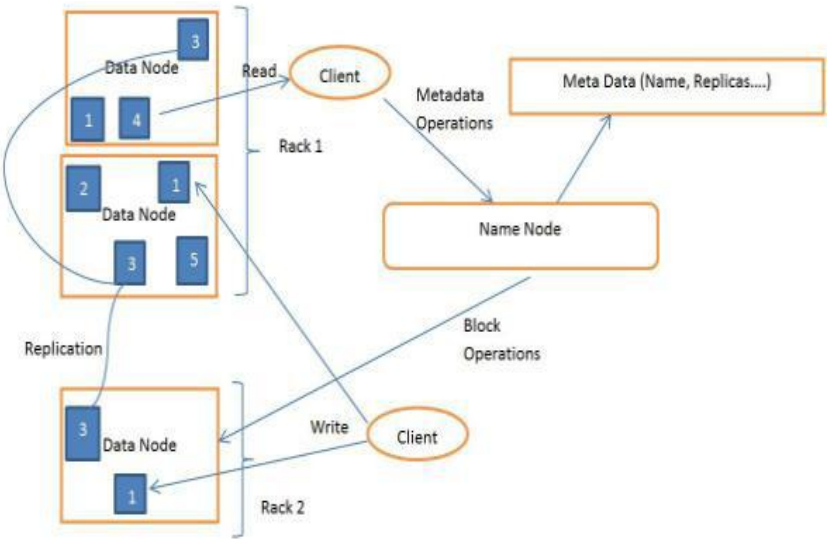


Fig (1) Hadoop Distributed File System architecture

B. Hadoop MapReduce

MapReduce is one of the parallel data processing programming framework designed for large scale data processing on cluster-based computing architectures. This was originally proposed by Google to handle large-scale web searching applications. This approach has been proved to be an effective programming approach for data mining, and search applications in data centers. The advantage is that it allows programmers to abstract from the issues of scheduling, parallelization, partitioning, replication and main focus on developing their applications. It consists of data processing functions: Map and Reduce.Parallel Map tasks are run on input data which is partitioned into fixed sized blocks and produce intermediate output as a collection of key value pairs. These pairs are shuffled across different reduce tasks. Each Reduce task accepts only one key at a time and process data for that key and outputs the results as <key, value> pairs. The Hadoop MapReduce architecture consists of one Job Tracker (Master) and many Task Trackers (Workers). The JobTracker receives job submitted from user, breaks it down into map and reduce tasks, assigns the tasks to Task Trackers, monitors the progress of the Task Trackers, and finally when all the tasks are complete, reports the user about the job completion. Each Task Tracker has a fixed number of map and reduce task slots that determine how many map and reduce tasks it can run at a time. HDFS supports fault tolerance of MapReduce computation by storing and replicating the inputs and outputs of a Hadoop job. Hadoop jobs have to share the cluster resources, a scheduling policy is to be used to determine when a job can execute its tasks.[8]

1.2 Job Scheduling In Hadoop

There are various scheduling algorithms these are given below.

A. FIFO(First in first out) Scheduler

Hadoop use FIFO (First in First out) algorithm in default scheduling algorithm. The jobs of all users are rsubmitted to only one queue. According to the priority level and the time sequence when they are submitted, the entire job queue are to be scanned, and then a firstly arrival job is selected to execute. FIFO is simple, the cost of entire cluster scheduling process is less. Although FIFO scheduling algorithm is simple but there are a lot of limitations. It is to be designed only for single type of job, so when multiple users run multiple types of jobs, performance will be relatively low.[6]

B. Fair Scheduler

The Fair Scheduler supports for priority based system, so if a pool has not gotten its fair share for a certain period of time, then the scheduler will kill task in pools running over capacity in order to give the slots to the pool running under capacity. Tasks are to be scheduled in an interleaved manner . As jobs have their tasks allocated to Task Tracker slots for computation, the scheduler tracks the deficit between the amount of time actually used and the ideal fair allocation for that job. As slots become available for scheduling, the next task from the job with the high time deficit is assigned to the next free slot. Over time, this has the effect of ensuring that jobs receive equal amounts of resources . Shorter jobs are allocated resources sufficiently to finish fastly. At the same time, longer jobs are guaranteed to not be starved of resources. Two locality problems were introduced from fair scheduler are: head-of-line scheduling and sticky notes[6]

C. Delay Scheduling

Delay Scheduling is an outcome of strictly implementation of fair sharing by compromising locality. To resolve the problem of the locality, Delay scheduling algorithm is to be proposed, in which a job waits for a limited amount of time for a scheduling opportunity on a node which has data for it. The main aim of Delay Scheduling is to statistically multiplex clusters while maintaining less impact on fairness and to achieve high data locality. This algorithm

is used to temporarily relaxes fairness to improve locality problem by asking jobs to wait for a opportunity of scheduling on a node with its local data. [9]

D. Capacity Scheduling Algorithm

The Capacity Scheduler was developed by Yahoo . In this number of queues are created each queue with the configurable number of map and reduce slots. Every queue has to be assigned guaranteed capacity. During the execution of jobs queues are to be monitored, if Its whole capacity is not used its excess capacity used for another queue . Capacity scheduling can set the priority of the jobs within a queue. So job with the highest priority have access to resources firstly after that the low priority jobs. There is a restrict access control on queues . These access controls are defined as per basis of queue.[4]

E. Hybrid scheduler based on dynamic priority

it based on dynamic priority in order to minimize the delay for not fixed length concurrent jobs.it is degined to relax the order of jobs to maintain data locality . it also provides a user-defined service level value for Quality of services. This algorithm is designed for data intensive workloads. It also tries to maintain data locality during execution of job . this improved response time by means of relaxing the strictly proportional fairness with the easy exponential policy model. This algorithm is fast and flexible. This also improves response time for multi-user Hadoop environments.[5]

F. Longest Approximate Time to End (LATE)

This algorithm used to try to improve performance of Hadoop by attempting to find slowest tasks by calculating remaining time of all the tasks. It ranks tasks by estimating time remaining and starts a copy of the highest ranked task that has a progress rate lower than the Slow Task Threshold. The main pros of this algorithm is robustness to not homogenity of node since only some of the slowest speculative tasks are restarted. But it can not break the synchronization phase b/w the map and reduce phases of map reduce framework, but only takes suitable action on appropriate slow tasks.[6]

G. SAMR(Self adaptive Map Reduce Scheduling)

SAMR mapreduce scheduling is designed which uses the previous information and find the slow nodes. The previous information is stored in each nodes in the format of Xml. It used to adjusts time weight of each stage of map and reduce tasks in accordance to the historical information. It decreases the execution time of mapreduce job and also improves the mapreduce performance in the heterogeneous environment. It defined fast nodes and slow nodes to be nodes which can finish a task in a small time and larger time than most other nodes. [4]

II. LITERATURE SURVEY

Divya S, Rini Mary Nithila I, Vinothini M, Kanya Rajesh R described in their paper about FIFO scheduler . the default FIFO scheduler is available where jobs are to be scheduled in first in first out order with support for other priority based schedulers also.

Sagar Mamdapure, Neha Papat,Munira Ginwala described in their paper about task scheduling algorithm such as fair scheduler and capacity scheduler. The aim of scheduling algorithms is to minimize the execution time of applications that running in parallel and also to solve three main problems of locality,synchronization and fairness.

Sukhmani Goraya, Vikas Khullar described that Management of Big Data is a Challenging issue. The MapReduce environment is the widely used key solution for data intensive jobs. We will analyze map reduce pipelining and along with processing of Map phase and Reduce phase. Core schedulers FIFO, Fair and Capacity Schedulers have been discussed. The Scheduler assigns Map Reduce task to the resources and there is a challenge to the scheduler to schedule the task in a way that it is scalable. Existing work shows the performance of the Hadoop depends upon input data and configuration of the cluster. In this paper, we have analyzed the execution time for data intensive jobs with increasing volume of the data set. We have also compared the execution time of the task with existing scheduler and our proposed method for the scheduler.

R.Thangaselvi, S.Ananthbabu, R.Aruna described that in their paper that Mapreduce technique is being used in hadoop for processing and generating large datasets with a parallel distributed algorithm on a cluster. A key benefit of mapreduce is that it automatically handles failures and hides the complexity of fault tolerance from the user. Hadoop uses FIFO (FIRST IN FIRST OUT) scheduling algorithm as default in which the jobs are executed in the order of their arrival. This method suits well for homogeneous cloud and results in poor performance on heterogeneous cloud. Later LATE algorithm is based on three principles i) prioritising tasks to speculate ii) selecting fast nodes to run on iii)capping speculative tasks to prevent thrashing. It takes action on appropriate slow tasks and it could not compute the remaining time for tasks correctly and can't find the real slow tasks. Finally a SAMR scheduling algorithm is being introduced which can find slow tasks dynamically by using the historical information recorded on each node to tune parameters. SAMR reduces the execution time by 25% when compared with FIFO and 14% when compared with LATE.

III. PROBLEM FORMULATION

The Hadoop implementation creates a set of pools into which jobs are placed for selection by the scheduler. By default, all pools have equal shares, but configuration is possible to provide more or fewer shares depending upon the job

type. The number of jobs active at one time can also be constrained, if desired, to minimize congestion and allow work to finish in a timely manner.

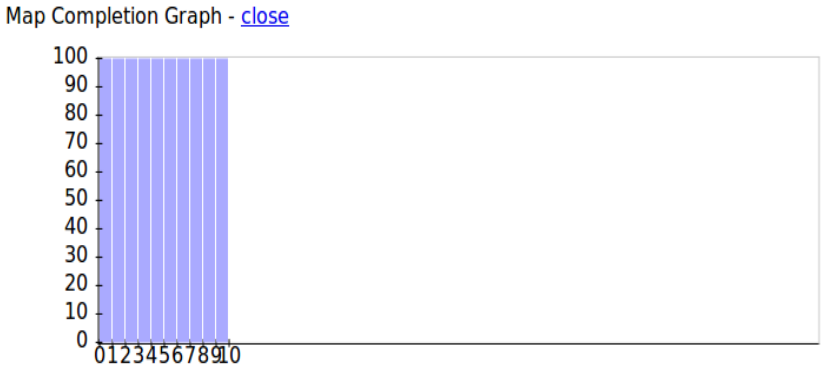
IV. RESEARCH METHODOLOGY

Step1: Flooding on hadoop datanode
Step2: Capturing the live traffic
Step3: copying that file to hadoop user
Step4: job assignment
Step5: map and reduce task
Step6:Analyze packet
Step7:calculate the flow of data on clients
Step8:calculate rate of data transfer by client.
Step9: collecting result

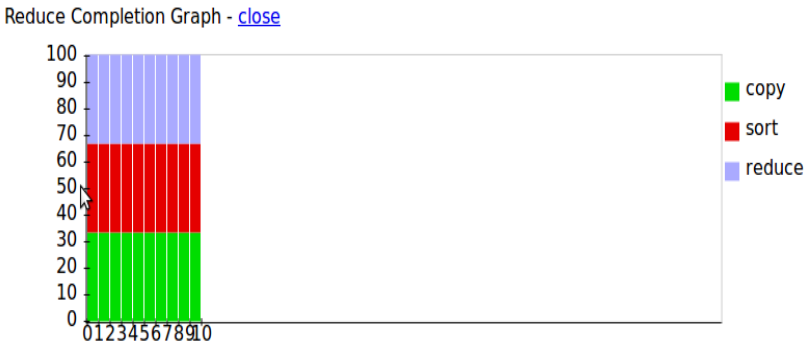
V. RESULT

We introduce the syn,udp,fin packets with the help of code attached to hadoop and then captured it with wireshark and then save the files after that copy these files in the home of hduser and then with the help of java code in hadoop calculate the no.of clients can transfer the data and then calculate the map reduce for each job after that find out the rate of transfer the data by each client.

The following map and reduce graph show the map and reduce task to calculate the no.of clients can transfer the data.

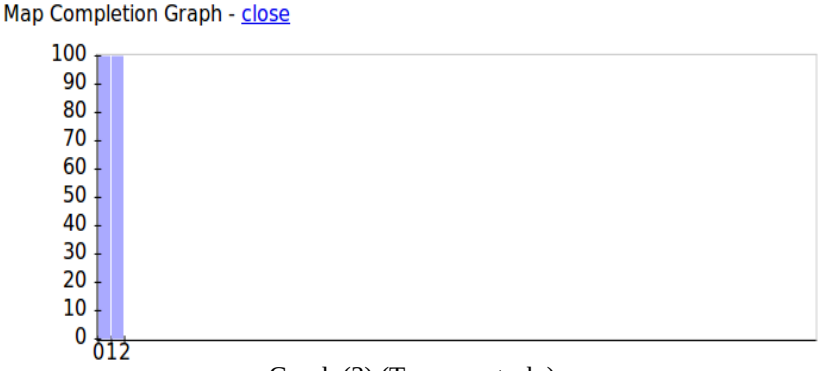


Graph(1) (ten map task)

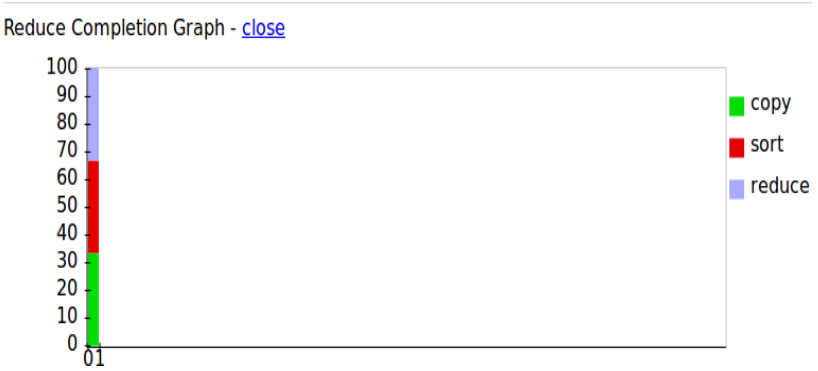


Graph(2) (ten reduce task)

The following map and reduce graph will show the the map and reduce task to calculate the the rate of transfer the data by each client.



Graph (3) (Two map task.)



Graph(4) (one reduce task)

VI. CONCLUSION

From the above we have concluded that hadoop is very effective to schedule the jobs scenario. Datanode of HDFS receives the blocks of data and scheduler for better job management in which adhoc query jobs can be executed with periodic jobs (for monitoring) in parallel that prevents the degradation in performance of distributed file system.this will show hadoop is fast to perform any jobs. The scheduling using unfair share increases the performance of the system.

VII. FUTURE WORK

We have implemented the execution of task to assign more or fewer shares depending upon the job type. For increasing the performance more than one jobs is executed in parallel by assigning unequal slots. Our future work is to introduce unequal shares of jobs based on the priority and we can also introduce unfair slots scheduling on multiple nodes to increase performance and minimize congestion on multiple node.

REFERENCES

[1] ZhuoTang ,Lingang Jiang , Junqing Zhou , Kenli Li and Keqin Li " A self-adaptive scheduling algorithm for reduce start time"

[2] M. Zaharia, D. Borthakur, J.S. Sarma, K.Elmeleegy, S. Shenker and I. Stoica, " Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling", In: Proceedings of the fifth European conferenccomputer systems. New York, NY, USA: ACM; 2010, p. 265–278.

[3] B.Thirumala Rao,Dr. L.S.S.Reddy"Survey on Improved Scheduling in Hadoop MapReduceinCloudEnvironments"International Journal of Computer Applications (0975 – 8887) Volume 34– No.9, November 2011

[4] Shyam Deshmukh"Survey on Task Assignment Techniques in Hadoop"International Journal of Computer Applications (0975 – 8887) Volume 59– No.14, December 2012

[5] Jilan Chen,Dan Wang,Wenbing Zhao"A Task Scheduling Algorithm for Hadoop Platform"JOURNAL OF COMPUTERS, VOL. 8, NO. 4, APRIL 2013

[6] Seyed Reza Pakize"A Comprehensive View of Hadoop MapReduce Scheduling Algorithms" International Journal of Computer Networks and Communications Security VOL. 2, NO. 9, SEPTEMBER 2014, 308–317 Available online at: www.ijcncs.org

[7] Divya S, Kanya Rajesh R, Rini Mary Nithila I, Vinothini M "Big Data Analysis and Its Scheduling Policy – Hadoop"IOSR Journal of Computer Engineering (IOSR-JCE)e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 1, Ver. IV (Jan – Feb. 2015), PP 36-40 www.iosrjournals.org

[8] Shivani Thakur,Rupinder Singh,Sugandha Sharma"Dynamic Capacity Scheduling in Hadoop"International Journal of Computer Applications (0975 – 8887) Volume 125 – No.15, September 2015

[9] Sreedhar C,N. Kasiviswanath,P. Chenna Reddy"A Survey on Big Data Management and Job Scheduling"International Journal of Computer Applications (0975 – 8887) Volume 130 – No.13, November 2015

[10] R.Thangaselvi, S.Ananthbabu, R.Aruna"An efficient Mapreduce scheduling algorithm in hadoop" International Journal of Engineering Research & Science (IJOER) [Vol-1, Issue-9, December- 2015]

[11] Sukhmani Goraya, Vikas Khullar"Enhancing Dynamic Capacity Scheduler for Data Intensive Jobs"International Journal of Computer Applications (0975 – 8887)Volume 121 – No.12, July 2015

[12] Sagar Mamdapure,Munira Ginwala,Neha Papat"Task Scheduling in Hadoop"Imperial Journal of Interdisciplinary Research (IJIR) Vol.2, Issue-1 , 2016ISSN : 2454-1362 , www.onlinejournal.in

