IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 3, March 2017

Responsive Job Scheduling for Map-Reduce in Hadoop Framework

Poonam Mahajan[1], Manish Patel[1], Amol Agarwal[1], Nikhil Raut[1], Devendra Gadekar[2]

Bachelors in Engineering, Computer Engineering, JSPM's Imperial College of Engineering and Research, Pune, India[1]

Professor, Computer Engineering, JSPM's Imperial College of Engineering and Research, Pune, India[2]

Abstract: Hadoop is a framework which is used to store and process large amount of data. Scheduling of Jobs is very much important to achieve high performance in Hadoop cluster. Hadoop scheduler is pluggable module which is used to manage the tasks for executions. Most commonly used schedulers are FIFO, Fair and Capacity scheduler. In this paper we have implemented Responsive Job Scheduler based on locality of Reference, it would add fair and capacity scheduler's job selection features to our algorithm. Data locality ensures that the map tasks will be executed on the node encompassing the input data. Though the proposed algorithm is designed specifically for map reduce framework and it can be very well implemented in Hadoop environment.

Keywords: Hadoop, MapReduce, Job Scheduler, Responsive Job Scheduling.

I. INTRODUCTION

Hadoop is a Open Source Framework by Apache. It provides distributed storage with computation across cluster of computers. Two core components in Hadoop are Hadoop Distributed File System (HDFS) and MapReduce. HDFS is Hadoop file system is used to store huge data in distributed manner [1].

Parallel processing of these huge amounts of data is necessary for fast processing of data sets, MapReduce is the programming paradigms that enables to process huge amount of data in parallel. MapReduce performs two tasks mapping task and reducing task.

Hadoop frameworks distributes all kinds of computation tasks to multiple computers for parallel processing of task. Its distributed approach achieves fault tolerance , fast processing , reliability , etc.

The Hadoop scheduling model is a Master/Slave (Master/Worker) cluster structure. The master node (JobTracker) coordinates the worker machines (TaskTracker). JobTracker is a process which manages jobs, and TaskTracker is a process which manages tasks on the corresponding nodes. [2]

There are three important scheduling issues in MapReduce such as locality, synchronization and fairness. There are many algorithms to solve this issue with different techniques and approaches.

Some of them get focus to improvement data locality and some of them implements to provide Synchronization processing. also, many of them have been designed to minimizing the total completion time.

With the number of data intensive jobs submitted increases, the load of the cluster also increases gradually. In order to manage load of the cluster and to provide proper resources of cluster to the running task a scheduling approach is needed to improve the overall cluster performance.[2]

II. RELATED WORK

There are number of scheduling algorithms which are available which helps Hadoop to improve its performance in different factors such as Data locality rate, Synchronization overhead, Fairness, Average response time, Job execution time and Average waiting time. Each of the traditional algorithm improves the performance in some of its factors.

R.Thangaselvi in [3], tunned the parameters using k means clustering technique and then assigning tasks to each node thus improving the performance of Hadoop in the heterogeneous . It implemented Lloyd's Algorithm.

Nenavath Srinivas Naik in[4], review some of the approaches of scheduling algorithms like LATE, SAMR and ESAMR, which have been aimed specifically to make the performance of MapReduce adaptive in heterogeneous environments is being showcasted.

Dharmesh Patel in [5], proposed scheduling by considering the load statistics of the TaskTracker into the Account . TaskTracker will send current load statistics by modifying heartbeat message. The load statistic information includes the CPU information, physical memory, swap memory, disk IO etc.

Radheshyam Nanduri in [6], proposed an approach which tries to maintain harmony among the jobs running on the cluster, and in turn decrease their runtime. In our model, the scheduler is made aware of different types of jobs running in the cluster

III.EXISTING SCHEDULER

A **First In First Out Scheduler**

The default scheduling algorithm that was integrated within the JobTracker was called FIFO. In FIFO scheduling, a JobTracker executes the jobs from a queue, on basis of their arrival .The task arriving earlier gets

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 3, March 2017

executed first . This schedule had no concept of the priority or size of the job .

B **Fair Scheduler**

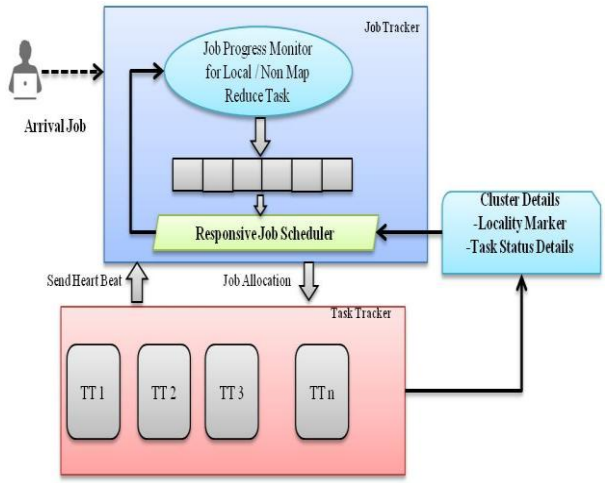
The Fair Scheduler gives an equal share of resources over time to each task. When there is a single app running , that app uses the entire cluster resources . When other apps are submitted, resources that free up are assigned to the new apps, so that each app eventually on gets roughly the same amount of resources.

The main objective is to create an equal distribution of computing resources among the users/jobs in the cluster. In case of multiple users, one pool is assigned to each user. The scheduler divides resources equally among these pools and jobs within a pool can then be scheduled based on priority, FCFS or FS basis.

IV. PROPOSED SCHEDULING ALGORITHM

In this paper we propose Responsive Job Scheduler based on locality of Reference .The proposed algorithm is designed for a Hadoop cluster that consists of the N= {n1,n2....n}nodes. Every node in the cluster is combination of resources including memory , disk , network connectivity , processor, etc .

Each cluster consists of {SN1, SN2, …. , SNn-1} SlaveNodes (SN) and single Master Node (MN). It is the responsibility of MasterNode to analyse available resources on SlaveNodes. The Master Node MN maintains a job queue JQ.



V. DETAILED DESIGN ANALYSIS

The terminologies used in the algorithm are:

a) MapReduce slots: It is defined as the total number of map tasks and reduce tasks that can be implemented parallely on a cluster node.

b) Locality marker: It is used to mark the slave nodes in order to ensure that each slave node gets an equal opportunity to grasp its local map tasks.

c) Local map task: It is a map task that is implemented on a slave node containing its input data .

d) Non-Local map task: It is a map task that is implemented on a slave node that does not contains its input data .

e) Heartbeat : It is a signal that slave nodes sends to the master node. It carries statistics about complete storage capacity, fraction of used storage capacity, amount of unused map slots, amount of unused reduce slots, and the amount of data transfers that are in progress.

The work flow of scheduling of tasks is as :

1.) The new task arrives in Job Queue (JQ)
2.) Job is divided into two - Map task and Reduce task.
3.) Every Map task is further divided into Local Map task (T_l) and Non-Local task (T_{NL})
4.) When a new Slave task is added , then it checks for for free slot count.

If free slot count is available is available , it sends heart - beat to master node

5.) The Master-node(MN). is responsible for scheduling jobs to Slave-node(SN).
6.) Scheduler checks for whether the task is local or non-local task.If it is Local Map task (T_l) , it gets allocated to slave node for execution and also locality marker is incremented by 1.

It continuously checks for sub-sequent jobs in JQ for (T_l)

If it does not find any (T_l) for current heart-beat , it sets the slave node's locality marker as 'zero'

7.) If it cannot find any (T_l) for next heart beat , it allocates (T_{NL}) from the JQ to slave node.

The data locality rate can be defined as:

$$\text{Data locality} = \frac{\text{no. Of local map task}}{\text{Total map task}}$$

The fairness of a job is calculated using

$$\text{fair share} = \frac{\text{job weight}}{\Sigma\,\text{job weight}} * \text{Capacity of taktracker}$$

The average response time of the total number of map tasks can be defined as :

$T_{avg} = R_l * T_{avg\,l} + (1 - R_l)\, T_{avg}^{\,n\,th}$

Performance =

$$\text{Map task} = \frac{\text{no .of pending map tasks}}{\text{no .of currently running map task}}$$

$$\text{Reduce task} = \frac{\text{no .of pending reduce tasks}}{\text{no .of currently running reduce task}}$$

VI. PERFORMANCE ANALYSIS

The proposed algorithm is evaluated using 2 factors: average waiting time of the job and utilization of resources in the cluster. We carry out experimental analysis using MapReduce WordCount job as benchmark in Hadoop 0.20.

We evaluate the performance of the proposed algorithm for best case, worst case, and average case . In best case,

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 3, March 2017

the job consists of all non-local map tasks whereas in worst case, the job consists of all local map tasks. In average case, job consists of several non-local map tasks and several local map tasks.
The experimental results show that the proposed algorithm reduces the average waiting time by 79% , further we aim to analyze the performance of the proposed algorithm based on earliest deadline first of the job.

VII. CONCLUSION

Using this proposed scheduling algorithm we may able to make more and proper utilization of resources of Hadoop cluster. This will help to decrease waiting of jobs in queue and will also reduce the execution time of job. This will be an important module which will help Hadoop to work efficiently and reliably.

ACKNOWLEDGEMENT

We thank our guide **Prof. Devendra P. Gadekar for** inspiring us to do this project, guidance, support and his comments in the manuscript.
We would also like to thank **Prof. Suryakant Bhalke** for continuous assistance for our project that greatly assisted the research.

REFERENCES

[1] Hadoop home page. http://hadoop.apache.org/.
[2] Supriya Pati and Mayuri A. Mehta , " Job Aware Scheduling in Hadoop for Heterogeneous
[3] Cluster " , 2015 IEEE International Advance Computing Conference (IACC)
[4] R.Thangaselvi , "] Improving the efficiency of MapReduce scheduling algorithm in Hadoop " ,
[5] Nenavath Srinivas Naik , A Review of Adaptive Approaches to MapReduce Scheduling in Heterogeneous Environments 978-1-4799-3080-7/14/$31.00@2014 IEEE
[6] Dharmesh Patel "Optimizing MapReduce Scheduling using Datanode Load Prediction "
[7] Radheshyam Nanduri, Nitesh Maheshwari, " Job Aware Scheduling Algorithm for MapReduce Framework" , 3rd IEEE International Conference on Cloud Computing Technology and Science Athens, Greece.
[8] Hadoop, "Hadoop fair scheduler" https://hadoop.apache.org/docs/r1.2.1/fair_scheduler.html
[9] "FairScheduler",https://www.ibm.com/developerworks/library/os-hadoop-scheduling.html
[10] A Comprehensive View of Hadoop MapReduce Scheduling Algorithms, "Fair Scheduling" http://www.ijcncs.org/published/volume2/issue9/p5_2-9.pdf
[11] R.Thangaselvi, S.Ananthbabu, R.Aruna." An efficient Mapreduce scheduling algorithm in hadoop", International Journal of Engineering Research & Science (IJOER), December- 2015
[12] C.He, Y.Lu, and D. Swanson, "Matchmaking: A new mapreduce scheduling technique," IEEE 3rd International Conference on Cloud Computing Technology and Science, December 2011.