Scientific Journal of Impact Factor(SJIF): 3.134

e-ISSN(O): 2348-4470
p-ISSN(P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 1,Issue 12, December -2014

SURVEY ON IMPROVE JOB PERFORMANCE USING MAPREDUCE IN HADOOP

Yash K. Merja[1], Prof. Rupesh G. Vaishnav[2]

[1]M.E. [Computer Engineering], Darshan Institute of Engineering & Technology, Rajkot, yashmerja@gmail.com
[21]M.Tech. [Computer Engineering], Darshan Institute of Engineering & Technology, Rajkot, upeshvaishnav0097@gmail.com

Abstract — *In recent years the different scheduling technique became a more attractive point mark to many researchers, but even though many achievement that have been made, then also still many trouble in this field. The main goal of this paper is to provide a better understanding of scheduling technique in MapReduce and analyze very important research directions in this field. There are lots of key techniques in MapReduce to improve the job performance in open source platform to choose as Hadoop. To address it, analyze and optimize the resource allocation in different aspect. The performance of Hadoop can be increased by proper resource management to the task in default scheduling technique. In Hadoop a program called map reduce is used for gathering data according to query. As use Hadoop is used for large amount of data therefore scheduling in Hadoop must be effective for better performance. So objective is to study and identify various scheduling technique, which are used to maximize job performance in Hadoop.*

Keywords*-Cloud Computing; MapReduce; Hadoop; Scheduling Technique; Data Locality; Deadline Constrain*

I. INTRODUCTION

Hadoop is much more than a highly available, unstructured data for data collection engine. An additional advantage of using Hadoop is that you can combine data storage and processing [1]. It can provide much needed robustness and scalability option to a distributed system as Hadoop provides low cost and reliable storage. Hadoop uses Hadoop Distributed File System (HDFS) for data storing and using MapReduce to processing a parallel data. It is designed to control a large amount of data, and provide permission to access this data to many users distributed across a network [2]. MapReduce is an attractive model for distributed computing, first presented by Google in 2004. All MapReduce job is compose of a certain number of map and reduce tasks. The MapReduce model for serve the multiple jobs contain of a processor sharing queue for the Map Tasks and a multiple server queue for the Reduce Tasks [3]. Hadoop allows the user to compose the job, submit it, control its execution, and query problem the state. Most of the job contains individual tasks, and all the tasks need to have a machine slot to run in Hadoop all scheduling and allocation purpose are made on a task and node level for both the map and reduce phases [4].

There are three important scheduling problems in MapReduce such as data locality, straggler problem and Deadline constrain. Data Locality is described as the distance between the input data node and task-assigned node. Straggler problem define node crashes, MapReduce runs its tasks in a different system. It's most important, if a node is available but then also it perform not better as we aspect, that condition known as straggler. Deadline constrains described as the issue of deadlines of job but it focuses on increasing system performance. Deadline required in Hadoop based data processing is done by job execution cost model that consist of different parameter such as map and reduce runtimes input data sizes, node level capacity and data distribution etc. As a result of the important problems and many more issue in scheduling of MapReduce, the scheduling is one of the most critical aspects of MapReduce. There are many algorithms to solve this problem with different techniques and methods. Some of the important issue to improve the data locality, straggler problem and deadline of the job has been designed to minimizing the total completion time.
.

II. BACKGROUND

2.1 Overview of Cloud Computing

Cloud computing has received important consideration in both academic and industrial communities in recent years. It vision of moving data storage and computing power away from local servers, over the network cloud, and into large data centers. Now it is playing an important aspect for processing huge volume of data in many areas.

Moreover, cloud computing accuse to the applications hosted by data centers and delivered across the Internet. Search from Google engine and Yahoo, Microsoft online video streaming from YouTube, purchases from Amazon and eBay and social networking from Facebook and Twitter are couple of prominent examples.

Furthermore, cloud computing also accuse to the hardware and software base that support those cloud applications. Cloud computing infrastructures compose on top of large scale clusters of commodity hardware machines. It produces satisfactory scalability and highly parallel execution environment for applications.

2.2 Overview of Hadoop

Hadoop implements master-slave architecture, in that master is named as NameNode and slave is named as DataNode. Namespace contains hierarchy of files and directories used for data storage. When file is created by user application, its split into blocks, each block is replicated and stored in DataNodes. In this case, info about the replicas numbers (number of block copies) and the mapping of replicas and blocks are stored in the NameNode. Additionally, all DataNode is in charge of managing storage attached to the node in which it is running on. Moreover, each DataNode handles the read operation, write, block creation, deletion and replication that come as instructions from the NameNode [5].

Besides NameNode and DataNodes, Hadoop cluster are contained as secondary NameNode (Backup Node), TaskTracker and JobTracker. JobTracker is detecting in the MasterNode and it is responsible of distributing MapReduce tasks to other nodes in the cluster. Additionally, TaskTracker runs local tasks distributed by the JobTracker, All slave in the cluster consist of one TaskTracker that can also run on MasterNode [5].
The overall architecture of Hadoop is illustrated in Figure 2.1.

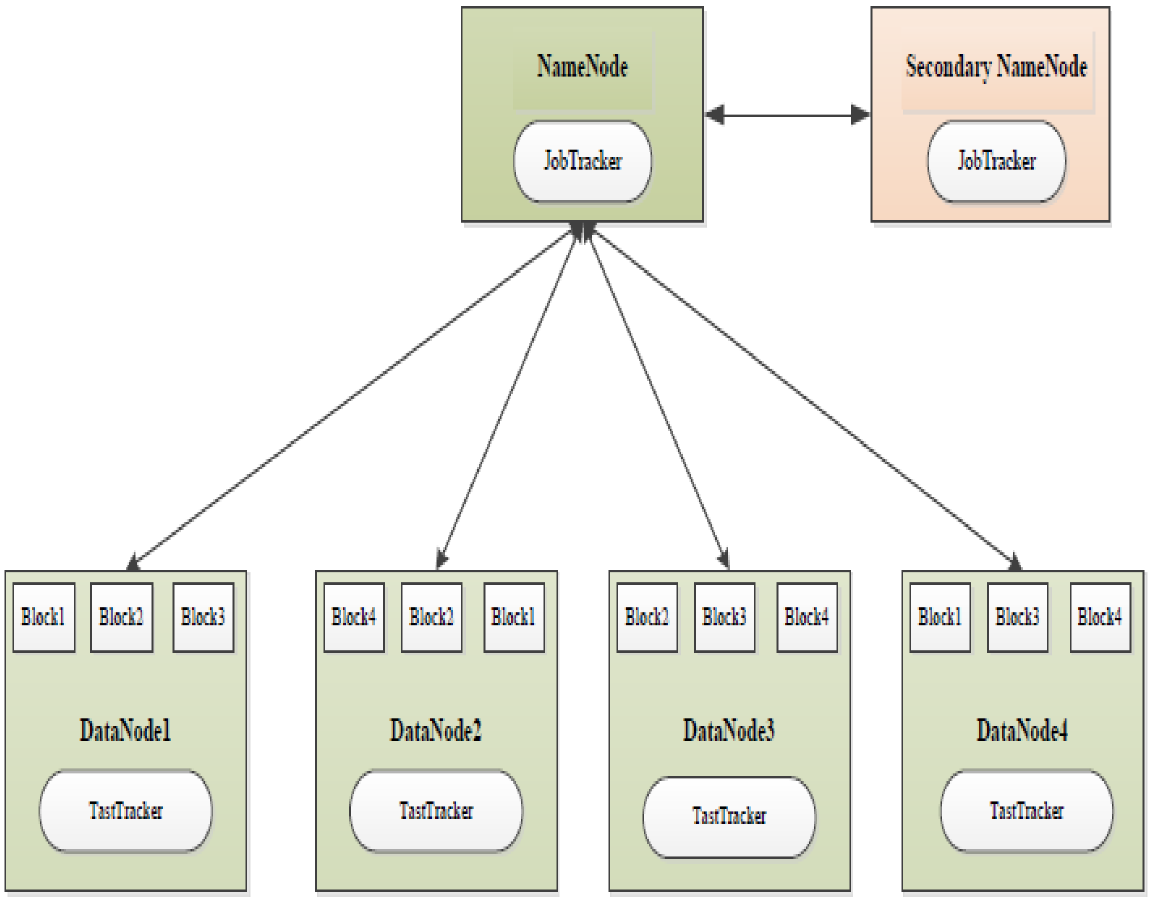

Figure 2.1: Hadoop Architecture

2.3 Overview of MapReduce

Hadoop MapReduce is a programming paradigm that processes very large data sets in parallel manner on large clusters. The basic idea behind the MapReduce is divided the input data set into chunks that will be processed by Map tasks in a parallel manner. The outcome of the all Map task is stored to direct as an input to the Reduce task. According to the definition of MapReduce can be categorize into two steps: Map task and Reduce task [6].

Map Task: In this process divided by itself into five phases: Read, Map, Collect, Spill and Merge.

- The Read Phase expressed by reading the data slot from the HDFS and then creating the input key-value.
- Map Phase is about executing the user defined map function to accomplish the map output data.
- Collect Phase is performs the collection of intermediate (map-output) data into a buffer before spilling.
- Spilling Process short the data and writing into local disk to create file spills.
- Marge Phase is last step of Map task in which all file spills into one single map output file [6].

Reduce Task: In this process divided into four phases: Shuffle, Merge, Reduce and Write Phase.

- The Shuffle Phase substitute the intermediate data (Map Output) from the Mapper slaves to a reducer's node and decompressing if needed.
- Merge Phase is performs the merging of the shorted outcomes that come from different Mappers to be directed as the input to the Reduce Phase.
- Reduce Phase executes the user defined function to produce the final outcome of data.
- Write Phase finally compresses, if needed and writes the final outcome to HDFS [6]

III. LITERATURE SURVEY

The important problems are that we define and more issue in scheduling technique of MapReduce, the scheduling is one of the most critical criteria of MapReduce. There are many algorithm can address these problem with different techniques and methods. Some of them focus on dynamic slot allocation and straggler problem for speculative execution. Also many of them have been design deadline constrain to minimizing the total job completion time. In this section we describe some of these algorithm techniques.

The computing Resources are considering into map and reduce slots, which are primary computing units and statically configure by administrator in advance. A MapReduce job execution has two main features: first, the slot allocation constrains that map slots allocate to map slots and reduce slots can be allocated to reduce task. There is immense different performance and system utilization for a MapReduce over differ slot configurations. So if we use each and every slot in both slot according to needs of node which affects the system utilization and performance. For that use a Dynamic Hadoop Slot Allocation (DHSA) [7] technique to increase the slot utilization for MapReduce.

Straggler Problem occur because of unavoidable run-time contention for processor, memory, network bandwidth and also other resources causing great affect on delay of the whole job. For that use a Speculative Execution Performance Balancing (SEPB) [7] to balance the utilization for single job as well as multiple jobs.

Data Locality increase slot utilization efficiency and performance achieve great output for MapReduce workloads. After all, there is often struck between fairness and data optimization in a shared cluster among many users. For that use a Slot PreScheduling technique to achieve great significant data locality at the expense of the load balance nodes.
User having specific job deadline so deadline are most important requirement which can improve the performance. MapReduce Task Scheduling algorithm for Deadline Constrains (MTSD) [8] has main goal on user's deadline constraints problem.

The Comparison Table of different Scheduling Technique in MapReduce

Scheduling Technique Name	**Categories**	**Advantages**	**Disadvantages**
DHSA	Slot Based Resource Model	Maximize the Slot Utilization	———
SPEB	Straggler Problem	Balance the Performance tradeoffs	———
Slot PreScheduling	Data Locality Issue	Data Locality	Load Balance Across Nodes
MTSD	Job Deadline Constrains	➢ Data Locality ➢ Deadline Constrains	———

IV. FACTORS AFFECTING JOB PERFORMANCE

To Increase job performance effective resource management is fundamental need in MapReduce to get the most value out of available resources. A resource in our analysis means a computational resource such as Memory, Disk Space, CPU or Network Bandwidth. We analyze the following most important issue in resource management in MapReduce:

1. Job Scheduling

2. Speculative Execution
3. Data Locality Issue
4. Task Assignment
5. Admission Control

4.1 Job Scheduling

Job Scheduling is the issue of deciding the arrangement in which a set of jobs is to be executed on a MapReduce cluster. The pattern of jobs could be driven by a user specific benchmark which usually includes minimizing response time of a job. Response time is the time from the submitting a job to its completion. Scheduling could also be impact on system owner's goals remarkably they consist of optimum resource utilization and servicing as many users simultaneously as possible [9].

4.2 Speculative Execution

Speculative execution in the case of MapReduce is about improving response time of a job by relaunching delay processing tasks by relaunching them on nodes with good resource availability. Crucially, speculative execution improves job response time in large and heterogeneous cluster [10]. There are two important issues involved in speculative execution:

- Deciding which of the currently executing tasks are progressing slowly relative to other tasks.
- Deciding a node on which a slow task must be relaunched.

Speculative execution try to rebalance on the WorkerNode and tries to improve response time by relaunching slow tasks on different TaskTrackers with many Resources. In this approach the Hadoop scheduler relaunches tasks slot that are processing slowly, compared to other running tasks. Slow tasks are replicated on machines with free task slots. This tries to achieve very good utilization when the job is to end. This also decrease job make span by decreasing runtime of the slow tasks slot. This also counters cause of overload by multiple task assignments on better machines.

4.3 Data Locality Issue

Data local execution is the idea of executing tasks as close as possible to their input data is embraced heavily in MapReduce. Speculative execution try to rebalance on the WorkerNode and tries to improve response time by relaunching slow tasks on different TaskTrackers with many Resources. The main reason behind this being that it is much cheaper to dispatch code to input data, than bringing input data to code. A good way, it reduces latency problem which may increase if input data needs to be reallocate over the network thereby significantly reducing response time [11]. The main issues are here: To discover the underlying network topology, and choosing and duplicate placement in order to increase data locality.

Data locality about executing tasks slot as close to their input data set as possible Hadoop tries to accomplish node level data locality. This is done to exploit the node level topology in typical data centers. A topology character gives the node location and distance between input split location and a node under consideration for task assignment. In the default scheduler tries to assign task to nodes whose distance with input data size is zero. Distance between a node and its input is zero if both reside in the same machine. Next nodes with distance one are considered these are the machines that are within the same server node. If not found such machines are found, and then Hadoop assigns task to non data local task i.e. the task whose input data are not in any of the system in the same node as the running of the task.

4.4 Task Assignment

In Task Assignment job scheduling deals with high level ordering of jobs, task assignment is all about choosing which task should be allow to a given WorkerNode. The task assignment should improve utilization of resources on the WorkerNode, and at the same duration anticipate overload. It depends on the current state of resources on the WorkerNode as well as the approximated output of assigning a task on the concerned node.

4.5 Admission Control

Admission Control is prevent overburden on a MapReduce cluster and to meet QoS requirements of already running jobs, jobs submission to a MapReduce cluster must be selectively accepted. Which jobs are accepted for execution depends on the goals of the system owner. Preventing overload of computational resources on the MapReduce cluster, and to maximize the value earned after successful execution of a job, in cases where the users are charged for using the MapReduce cluster.

All the above issues are complicated by the fact that a MapReduce cluster generally comprises of cluster of commodity off the shelf hardware. Hence any resource management component must take into consideration dynamism in the availability of resources and the necessity of the application as well. It gives to user requirements of deadlines and guarantee of execution should also be considered while designing a result for the above issue.

V. PROPOSED WORK

There are lots of scheduling technique are available to improve job performance but all the technique have some little limitation so any one technique cannot overcome that particular parameter in which they effecting the performance to whole system. Like data locality, fairness, load balance, straggler problem and deadline constrains. All the technique has advantages over any other technique so if we combined or interchange some technique then the result will be even much better than the individual scheduling technique.

Slot PreScheduling and MapReduce task scheduling algorithm for deadline constraints (MTSD) are both technique are used for improve data locality but MTSD algorithm handle data locality as well as job deadline constrains according to measuring a node's computing capacity have more efficient improvement in job performance for whole system. So when we interchange the Slot PreScheduling technique to MTSD algorithm in Dynamic MR slot allocation optimization framework for MapReduce cluster that will be additionally add the deadline constrain mechanism to improve the job performance of whole system and also much better data locality also achieve the attractive performance in Dynamic MR.

VI. CONCLUSION

For the better improvement in job performance many scheduling technique are available but all the technique have some limitation and one technique can't provide the better outcome so for the improvement of the performance we need to combined two or more scheduling technique they have advantage over each other technique. MTSD algorithm provide data locality as well as additional user's specify job deadline constrain as compared to Slot PreScheduling.

REFERENCES

[1] A. N Nandakumar and Y. Nandita, " A Survey on Data Mining Algorithms on Apache Hadoop Platform", International Journal of Emerging Technology and Advanced Engineering, Vol. 4, NO. 1, January 2014, pp. 563-566.
[2] Z. Tang, L. Jiang, J. Zhou, K. Li, and K. Li, " A self-adaptive scheduling algorithm for reduce start time ", Future Generation Computer Systems, 2014.
[3] K. Morton, M. Balazinska and D. Grossman, " Paratimer: a progress indicator for MapReduce DAGs", In Proceedings of the 2010 international conference on Management of data, 2010, pp.507–518.
[4] Lu, Wei, et al. "Efficient processing of knearest neighbor joins using MapReduce ", Proceedings of the VLDB Endowment, Vol. 5, NO. 10, 2012, pp. 1016-1027.
[5] T. White, *Hadoop: The Definitive Guide*, O'Reilly Media, 2010
[6] H. Herodotu, "Hadoop Performance Models", Computer Science Department at Duke University, 2011
[7] Shanjiang Tang, Bu-Sung Lee, Bingsheng He, "*DynamicMR: A Dynamic Slot Allocation Optimization Framework for MapReduce Clusters*", IEEE Transaction on Cloud Computing, 2014
[8] Zhuo Tang · Junqing Zhou ·Kenli Li · Ruixuan Li, " A MapReduce task scheduling algorithm for deadline constraints" Springer Science + Business Media New York, 2012.
[9] Suman Arora, Dr.Madhu Goel , Head of Deptt " Survey Paper on Scheduling in Hadoop" Volume 4, Issue 5, ISSN: 2277 128X, ijarcsse May 2014.
[10] Jeffrey Dean and Sanjay Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In Proceedings of the 6th Symposium on Operating Systems Design and Implementation, pages 137–150, 2004.
[11] Kavitha Ranganathan and Ian Foster. Simulation studies of computation and data scheduling algorithms for data grids. Journal of Grid Computing, 1(1):53–62, March 2003.