Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 9, September 2015, pg.105 – 112

RESEARCH ARTICLE

SLOT ALLOCATION AND LOAD BALANCING TECHNIQUE FOR MAPREDUCE

Mr. Kedarnath[1], Ms. Anjana Sharma[2]

Student, M.Tech (Computer Science and Engineering) New Horizon College of Engineering, Bangalore, India[1]

Assistant professor, Computer Science and Engineering department, New Horizon College of Engineering, Bangalore, India[2]

kedarnath.ssk@gmail.com, anjana.nhc@gmail.com

Abstract-DynamicMR technique is used to allocate slots dynamically and to optimize the resource utilization. This technique consist of three modules they are slot allocation, speculative performance balancing and slot prescheduling. These three techniques suffers some problem. Those problems are addressed in this paper. Slot allocation has the problem of allocating the slots for the task when there is no free map and reduce slots are not available this can be solved by establishing the communication between two cluster. The second method suffers with the problem of speculative task this problem is solved by investing the speculative task. The third technique suffers from the load balancing issue this problem is resolved by finding the duplicate copy of the data for computation.

Keywords- Hadoop, MapReduce, Job Tracker, Task Tracker, Speculative Task, Fair Scheduler

I. INTRODUCTION

1.1 Overview

With advances on mobile devices and wireless communication technologies, the demand for mobiles to run bigger applications (like which run on desktop computers) is on increase. Since mobile devices, even for the modern carrying devices that is mobility devices are concerned with size and weight, their resources for doing task and collaborations are limited checking with their desktop siblings. Therefore, it makes more sense for more weight

task to run in more powerful machines. There is a lot of advancement in the technology such as many other social networking sites which are regularly used by the many number of customers on daily basis. Which generates the huge data from day to day in order to store such huge data number of server storage is required so this type of data is stored in distributed servers. This huge data is known as big data. Big data is a data which cannot be stored or processed on single storage node or single machine. Big data size may varies from one terabytes to several hundred terabytes.

Big data processing can be done in traditional approach if there is any application which requires the big data to provide the solution to user query, in order to do that the data required for processing the request has to be brought at the computing node then the query is imposed on the computing node. This has certain challenges it is very difficult to transfer such a huge data to the computing node will need to have very high speed networks and the cost required to transmission of data is also high. So simplify this problem the new framework is introduced known as Hadoop.

Hadoop:

Hadoop is a framework of tools which is used for processing the big data. Hadoop has many tools which facilitates the processing of big data without any hindrance. Hadoop supports the local computation of the data in order to avoid the data transmission cost. Hadoop says transferring application input data to the computing node will improves the performance and also saves the data transmission cost. In Hadoop the input data is sent to the computing node across various distributed servers where its operation has to be done. Hadoop directs the all distributed server to process single job as a many tiny tasks. To do this Hadoop uses very strong programing technique which is known as MapReduce. MapReduce is algorithm which actually carries the operations on the locally available data based on the data which is received by the user application and finally gives the end result back to the user.

Hadoop has two components they are Hadoop distributed file system and MapReduce program. Hadoop distributed file system is used for local computation. The data is stored in the HDFS is accessed for local computation. In HDFS data is stored in block forma. The size of the block vary from 64 byte – 128 byte. When there is any operation has to be done the Hadoop will fetch the data from these block. This Hadoop distributed file system helps for the local computation of the data.

MapReduce

MapReduce is software program which is used for processing the task given by the Hadoop MapReduce software has Map function and Reduce function. Map function is used

for the map task. Map task is task in which the larger job is split into multiple small jobs known as task. Map task has its own task to perform and similarly for reduce task. Map task returns the key value pairs. These pair is received by the reducer phase and which will combine the appropriate result together and gives back to the user.

Hadoop is used for processing big data. MapReduce plays very important role in processing the job given to Hadoop. MapReduce version 1 includes some defects such as not optimized resource allocation which affects the performance of the Hadoop. There is a technique called dynamic Hadoop slot allocation framework (DynamicMR) which resolves the barriers of slot based MapReduce version 1 to improve the performance of MRV1. This project is undertaken to improve the performance of slot based MapReduce by identifying and optimizing resource allocation.

Hadoop consist of Job Tracker and Name node which are responsible for processing the job. In Hadoop there should be only one job tracker and only one name node. Job tracker divides the large complex task into smaller manageable piece of jobs known as task. Task tracker is responsible for running its task. After completing the task, task status is returned to the job tracker which later updates the system resource and slots. Name node contains the information about data block and corresponding data node. Name node contains the information about all the data nodes.

1.2 Problem Statement

We have the following observations:

I. There are significantly different performance and system usage for a two stage workload under different slot settings.

II. Even under the optimal map/reduce slot settings, there can be many unused reduce (or map) slots, which strongly mitigates the system usage and performance. Secondly, due to unavoidable run-time contention for processor, memory, network bandwidth and other resources, there can be straggled map or reduce tasks, causing significant delay of the whole job.

III. Thirdly, data locality maximization is very important for slot utilization efficiency and making more number of task in less time of workloads. However, there is often a conflict between fairness and data locality optimization in a shared Hadoop cluster among multiple users.

II. EXISTING SYSTEMS

i. DynamicMR:

Previously DynamicMR is the technique used for slot allocation and load balancing for the MapReduce framework.

DynamicMR technique consist of the following three modules they are listed below,

- Slot allocation.
- Speculative Execution Performance balancing.
- Slot Prescheduling.

Disadvantages of DynamicMR

- This technique is unable to allocate the slots to task when there is no idle map and reduce slots available in the cluster.
- If the task not completed within the given period of time task is going to be killed or suspended this degrades the performance of the whole system.
- Load balancing and fairness cannot be achieved at the same time. If load balancing is taken care it will cause an impact on the fairness. If fairness is considered it may impact on the load balancing.

ii. Multi-Threading

Multithreading concept is used for the MapReduce task which handles the task efficiently.

Advantages

- Handles the multiple job in heterogeneous environments.
- Improves the performances for all kind of workloads.

Disadvantage

- Pre configuration.
- Speculative task.
- Load balancing.

III. PROPOSED SYSTEM

This system is proposed based on identifying three problems encountered in the DynamicMR technique. The problems are load balancing, slot allocation and load balancing. These problems are resolved by using following modules of proposed system.

- Allocation model
- Speculation model
- Load Model

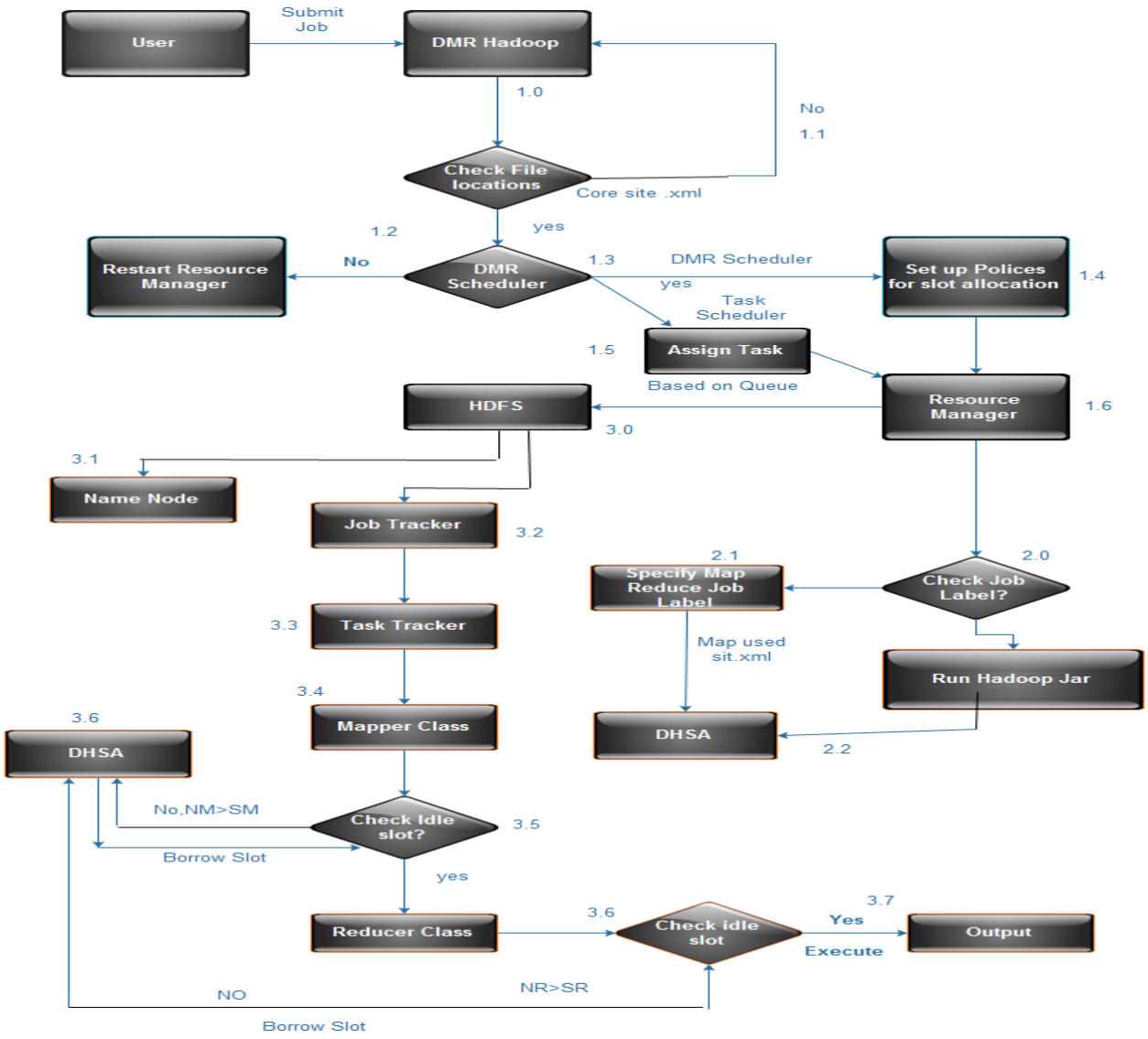

Fig 3. High level architecture of the proposed system

The above flow chart shows the high level design of the hadoop system with the DynamicMR this flow chart do not shows the model steps but helps to gain the understanding of the execution of the task. The problem which occurs in the DynamicMR are resolved by using below given model description.

Allocation Model:

In DynamicMR there is problem with allocating slots to the task when there is no idle slots available in cluster. This problem can be solved by using this approach.

The slot allocation procedure for the map task.

Step 1: if there is pending map task and idle slot

Step 2: assign Map task

Step 3: else if there is pending map task and idle reduce slot go to step 2

Step 4: if there is no is no idle map slots and reduce slots for pending map task

Then send request to nearer name node (nearer node is one which has less response time)

For idle slots

Step 5: if there is free slot in the nearer node allocate idle slots for the map task.

In the same way slots for reduce task also allocated.

Speculation Model:

In this model the speculation task is identified and it is rescheduled in the next cycle of the scheduler. If there is problem with the resource configuration then also the same kind of problem will occur result of which task will not complete in the second cycle of the scheduler, the speculative task remains to be speculative. This problem can be solved by using the following procedure.

Step 1: Identify the speculative task

Step 2: If the task is running after 2 heart beats then kill the current task

Step 3: Schedule the equivalent back up task.

Step 4: Schedule the failed/ killed task in the next cycle.

In order to finish the task in the next cycle we have to certain background analysis before scheduling the failed/killed task.

Steps for scheduling the failed/killed task

Step 1: obtain the configuration of failed task.

Step 2: identify the resource requirement for the failed task.

Step 3: Allocate twice of the required resource for the task.

Load Model:

In this model the data locality and load balancing along are addressed without affecting the fairness of the system. Load balancing can be achieved by assigning task to the node which has less load factor. Data locality is achieved by finding the duplicate copy of the required.

Procedure for the data locality and load balancing

Step1: calculate the load factor for each node in the cluster

Step2: assign the task to the each node in the cluster.

Step3: if there is any pending task to be scheduled then

Step4: find out where data is located.

Step 5: if data is located on the same node and which is has less load factor

Then scheduled task on that node

Step6: if the data is located on the current node but if node has highest load factor

Step7: Then find out the duplicate copy of the data (hadoop maintains three copies of data)

Step8: calculate the load factor for that node on which and calculate the response time of the node

Step9: if load factor and is less and response time is minimum than previous node then initiate the Task instance in that node.

IV. CONCLUSION

The proposed system easy to implement. Cheaper it can be implemented without removing existing configuration of the system. Using this proposed system,

- Speculative task can be handled much more efficiently.
- Load balancing is achieved without effecting the fairness.
- Slot allocation and utilization is enhanced.

REFERENCES

1. F Ahmad, S. Y. Lee, M. Thottethodi, T. N. Vijaykumar. *PUMA: Purdue MapReduceBenchmarks Suite*. ECE Technical Reports, 2012
2. G. Ananthanarayanan, S. Kandula, A. Greenberg, I. Stoica,Y. Lu, B. Saha, and E. Harris, Reining in the outliers in map-reduce clusters using mantri, in OSDI'10, pp. 1-16, 2010.
3. Apache Hadoop NextGen MapReduce (YARN).

http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html.

4. C. H. Bennett and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing," in Proc. IEEE Int. Conf. Compute., Syst. Signal, Bangalore, India, Dec. 1984, pp. 175–179.
5. J. Chao, R. Buyya. MapReduce Programming Model for .NET-Based Cloud Computing.In Euro-Par'09, pp. 417-428, 2009.
6. Q. Chen, C. Liu, Z. Xiao, *Improving MapReduce Performance Using Smart Speculative Execution Strategy*. IEEE Transactions on Computer, 2013.
7. J. Dean and S. Ghemawat. *MapReduce: Simplified Data Processing on Large Clusters*, In OSDI'04, pp. 107-113, 2004

