International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 04 Issue: 12 | Dec-2017 www.irjet.net p-ISSN: 2395-0072

Performance Improvement of Heterogeneous Hadoop Cluster Using Ranking Algorithm

Shivani Soni[1], Nidhi Singh[2]

[1] *M.tech Scholar, Dept. of Computer Science Engineering, L.N.C.T College, Bhopal (M.P), INDIA*
[2] *Professor, Dept. of Computer Science Engineering, L.N.C.T College, Bhopal (M.P), INDIA*

ABSTRACT: *Enhancing technologies increases the use and growth of information technology. As we can see the growth of data which is increasing rapidly in every single minute. This exponential growth in data is one of the reason for the generation of rapid data. Stored data is processed to extract worth from inaccurate data to form a way for the parallel and distributed processing for Hadoop. All the nodes in Hadoop are assumed to be in homogeneous nature but it is not same as it looks like, in cloud different configuration systems are used which represents it logically. So data placement policy is used to distribute data on the basis of power of node. Dynamic block placement strategy is used in Hadoop, this strategy work as the distributing input data blocks to the nodes on the basis of its computing capacity. The proposed approach balance and reorganize the input data dynamically in accordance with each node capability in a heterogeneous nature. Data transfer time is reduced in the proposed approach with the improvement in performance. Block placement strategy, page ranking algorithm and sampling algorithm strategies are used in the proposed approach. The data placement strategy used works as decreasing the execution time and improving the performance of the clusters which are of heterogeneous nature. Big data are handled using Hadoop. Small files are handled using applications on Hadoop so that the issue of performance can be reduced on the Hadoop platform. Better performance improvement is shown in the proposed work.*

Keywords: Hadoop; block placement strategy; page ranking algorithm; sampling algorithm

1. INTRODUCTION

In the year 2006, Doug Cutting and Mike Cafarella, developed Hadoop as a framework which is an open source computing and processing of large datasets in distributed environment. System failure and data loss can be reduced in the proposed work. Failure of any node does not matters because of the availability of thousands of interconnected node. Thousands of data and its frequent transfer is tackled among the nodes. Big data is handled by the Hadoop.

Hadoop is widely known because of its increase in popularity and handling of big data. To achieve better performance several techniques are used.

1.1 Overview of Hadoop:

The Google Map Reduce programming model implemented the open source framework called Apache Hadoop. Hadoop distributed file system HDFS and Map Reduce model are the two key parts of the Hadoop framework. HDFS stores application data for Map Reduce and map Reduce process the data using logics on map Reduce to store data on HDFS. Versions available of Hadoop are Hadoop 1.x and Hadoop 2.x. YARN is the improvement feature of Hadoop 2.x. Node Manager and Resource Manager are the two compositions of YARN, where both works differently. For deploying and managing resources, Resource Manager is responsible and for managing and reporting data node status to resource manager, Data node is responsible.

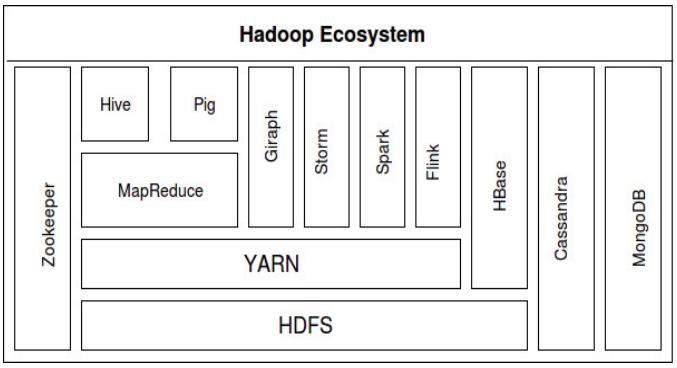

Figure 1. Hadoop Ecosystem

1.2 Big Data:

The collection of massive amount of data is called Big Data. This data can be in any form either structured or unstructured, relational or non-relational. Unstructured data is the audio, video, text, image or any different pattern of data. In recent years, Big data has become very popular in several different fields. This is a big opportunity in business field. Large transmission and communication of data generates large data from various sources. The need of data mining algorithm is required to process big data. Earlier production of large data is responsible due to corporate world but in recent years users have become responsible for its data.

1.3 Dynamic Block Placement Strategy:

Dynamic Block Placement Strategy works on two basis Homogeneous cluster and Heterogeneous cluster.

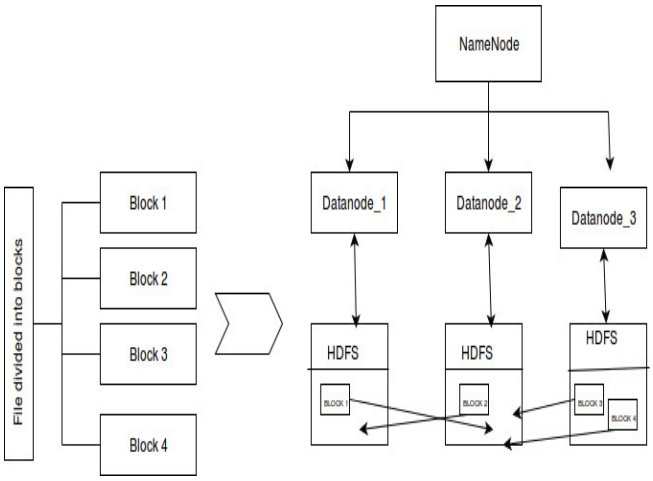

Figure 2. Block Placement Strategy

Homogeneous Cluster:

Depending upon the availability of space in a cluster data is distributed among the nodes in homogeneous cluster. Hadoop has a feature of balancing the data, this functionality of balancing is called Balancer, which balance the data before running the applications. Whenever conditions occur like on any node large data is accumulated then in this case balancer is an important functionality. Replication is an important function, balancer is responsible for it to care for replications. It is the key feature in data movement.

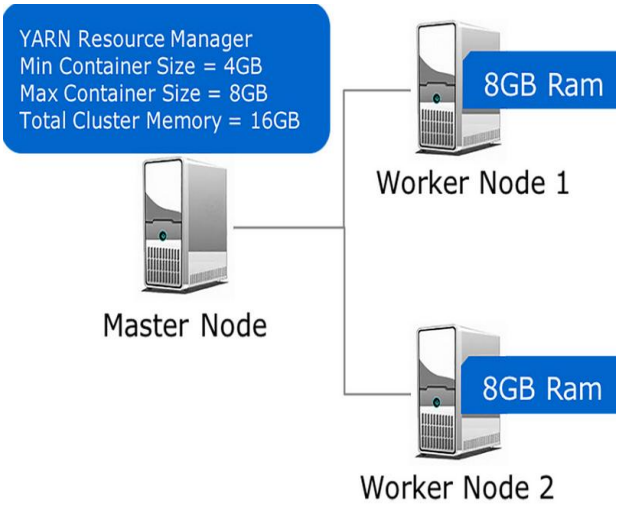

Figure 3. Homogeneous Cluster

Heterogeneous Cluster :

Data is transferred from one node to another node ideally in an heterogeneous environment. The faster node faces overheads while processing and data transfer. This results in exploring of the issues comes at the time of data transfer. Data placement policy explores the arising consequences. And this all occurs in the heterogeneous environment. Implementation of data placement policy provides the details of better goals.

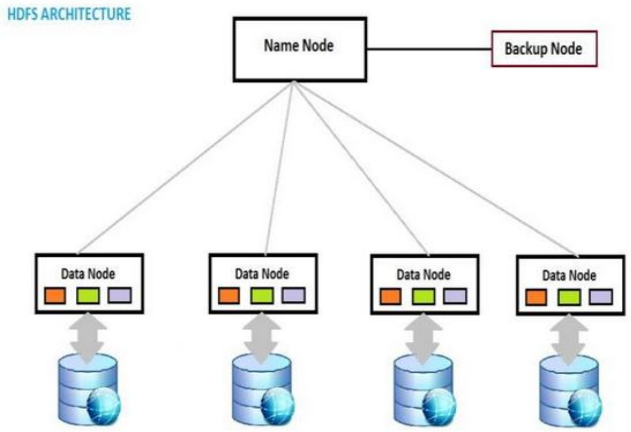

Figure 4. Heterogeneous Cluster

2. RELATED WORK

Many of the author stated and researched about the Big data, deal with them and also checks there functionality to work on homogeneous and heterogeneous cluster of data.

Jeffrey Dean et al. In [1] described about Hadoop, Hadoop process terabytes of data which is of large amount. The Apache group written Hadoop using java technology. It works as parallel processing to process large clusters. Hadoop is attractive and open source framework. It process the data and replicates it in reliable manner. It is designed in a manner to run commodity cluster. Low cost low performance working in parallel is preferred by commodity computing. HDFS is an Apache project for Hadoop, it is distributed, low-cost and have high fault-tolerance file system. It is convenient for large data and provides with high throughput. Its deployment is not costly. Single name node in each cluster maintains file system of Meta data and application data is stored by multiple nodes. Map Reduce analyzes large data and advantageous for various organization. Machine learning, indexing, searching, mining are some map Reduce applications. Traditional SQL are used to implement these applications. Also helps in data transformation, parallelization, network communication and handling fault tolerance.

Andrew Wang et al. In [2] explained about HDFS which is a distributed file system. It stores files across cluster node redundantly for the purpose of security. HDFS divides files into blocks and replicates them depending on the factor of replication. The block placement policy is the default in HDFS, it works as distributing blocks across cluster node. Many conditions like unnecessary load on cluster can be possible at any time which results in reducing the overall performance of cluster.

Konstantin Shvachko et al. In [3] proposed about block placement which plays a vital role in performance and data reliability terms. Reliability, availability and network utilization are improved by this data block placement strategy. At the time of creation of new block, the first replicated block is assigned in first location of the block. And the other replicates will be assigned randomly on different nodes keeping in mind that only two replicates should be placed in a single rack. Name node provides data node for HDFS, which helps in reducing network traffic and improves performance.

Fang Zhou et al. In [4] describes that application master generates input splits in a Hadoop map Reduce. One input split is generated for a small file. One map container can use only one input split, which explains that number of input splits and number of map containers are equal, which is the issue because it creates many map container for a small file. If many containers are created then it require many processes, resulting in many overheads. Similar overheads are generated for reduce container

3. PROBLEM DOMAIN

Hadoop handles the big data, which is now become a great deal to manage because of generation of data in every single minute. Big data is also the opportunity for business. But here we are looking forward on the issue of homogeneous and heterogeneous cluster.

Homogeneous cluster where clusters nodes are of similar form means they are homogeneous but load alloted on every cluster are not similar. This overloading of any cluster decreases the performance of cluster node and also reduces overall performance of outcome.

Similar to heterogeneous cluster, where data nodes are of different sizes and Hadoop works as distributing equal amount of load on the nodes. In this case if one node of larger size is experiencing low load then in that case that particular node completes it work firstly but it waits for the other node to complete their work because all these nodes after processing computes on the result, so for it, it waits for other nodes which results in reducing performance and increasing computation time.

Problem in it comes as, there are number of cluster nodes of different sizes like 2GB, 4GB, 6GB and 8GB. For example these four nodes are taken and the master node here works as dividing all the nodes with the number of nodes. Means the master node will divide those nodes with four.

In this mitigation approach, we are working on the issue of performance and computation time and will be achieved in our solution domain.

4. SOLUTION DOMAIN

The solution provided to the above problem will be described as using page ranking algorithm and sampling algorithm.

Where page ranking algorithm works on the basis of frequency. The one who is having better frequency will run first. Page ranking is used for ranking purpose and on the basis of frequency, ranking is allotted. Weight and frequency is calculated using page ranking algorithm and we used it in work.

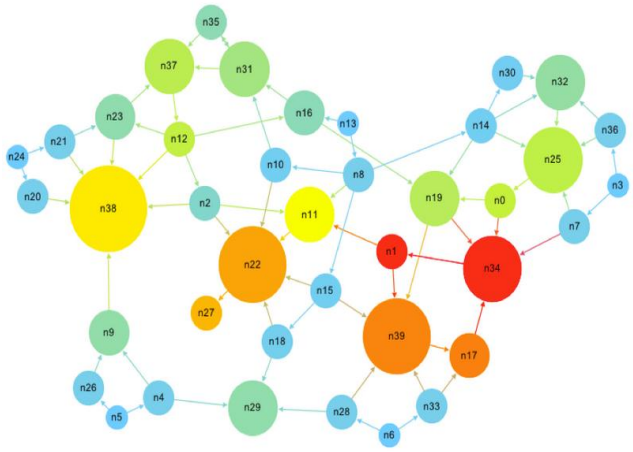

Figure 5. Page Ranking

Here, heterogeneous cluster is used and its performance is increased using ranking algorithm which depends on the frequency of occurrence. Whose frequency is maximum will be run first.

And sampling algorithm works as randomly selecting the nodes instead of mentioning all the possible samples. Probability of selecting is the sum which is equal to the sample size of data n. So, will result in increase in performance with reducing computation time and distributing the overall load in equal to all the data nodes.

5. RESULT ANALYSIS

Result analysis of the proposed work describes that the time required in proposed work is less then the time required in existing work. The below table shows the data of variant file sizes at different time and time is taken in second.

TABLE 5.1. Comparison table of existing work and proposed work

File Sizes	Original System	Proposed System
500 MB	249 sec	101 sec
1 GB	478 sec	226 sec
1.5 GB	633 sec	337 sec
2 GB	815 sec	462 sec
2.5 GB	1052 sec	560 sec
3 GB	1258 sec	652 sec

The below graph shows the graph representation of proposed work for different file size at different time.

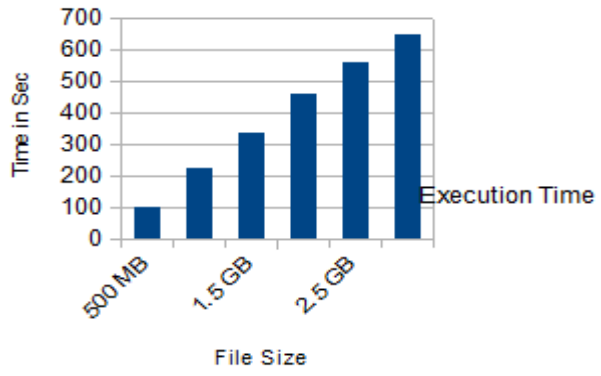

Figure 6. Graph representing the proposed work for different file size

The below graph shows the comparison of existing work with proposed work where the time taken for execution of variant file sizes in proposed work is less in comparison to existing work. Existing work requires more execution time.

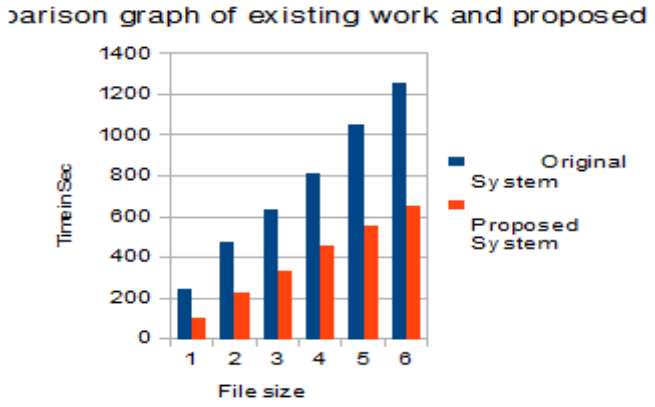

Figure 7. Comparison graph

6. CONCLUSION

The mitigation approach concluded that the data nodes can be of any size or of same size should not be overloaded and each node should assign the equal amount of load depending on there size which will result in increasing performance.

This paper attempts to improve performance of heterogeneous cluster in Hadoop using Ranking algorithm which works on the basis of frequency, the one who is having maximum frequency will be executed and run first. It reduces the computation time and improving performance.

7. FUTUTRE WORK

Depending on the allotment of work load on every node this issue of performance arises. In future implementation performance can be more improved using different techniques.

8. REFERENCE

1. Jeffrey Dean and Sanjay Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM, vol. 51, no. 1, pp. 107113, 2008.

2. Andrew Wang. Better sorting in Network Topology pseudo Sor tBy Distance when no local node is found. https://issues.apache.org/jira/browse/HDFS-6268, 2014. [Accessed 28-April-2014].

3. Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler, The hadoop distributed file system, in Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on. IEEE, 2010, pp. 110.

4. Fang Zhou, Assessment of Multiple MapReduce Strategies for Fast Analytics of Small Files, Ph.D. thesis, Auburn University, 2015.

5. Fang Zhou, Hai Pham, Jianhui Yue, Hao Zou, and Weikuan Yu, Sfmapreduce: An optimized mapreduce framework for small files," in Networking, Architecture and Storage (NAS), 2015 IEEE International Conference on. IEEE, 2015.

6. F. Ahmad, S. Chakradhar, A. Raghunathan, and T. N. Vijaykumar, "Tarazu: optimizing mapreduce on heterogeneous clusters," in Proc. of Int'l Conf. on Architecture Support for Programming Language and Operating System (ASPLOS), 2012.

7. M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in Proc. of USENIX Symposium on Operating System Design and Implementation (OSDI), 2008

8. H. Herodotou and S. Babu, "Profiling, what-if analysis, and cost based optimization of mapreduce programs," in Proc. Int' Conf. on Very Large Data Bases (VLDB), 2011