Scientific Journal of Impact Factor (SJIF): 5.71

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

International Journal of Advance Engineering and Research Development

Volume 5, Issue 04, April -2018

HETEROGENEITY BASED FAIRLY DATA DISTRIBUTION IN HADOOP ENVIRONMENT

Ramchandani Hema Megharajbhai [1], Viral Parmar[2]

[1]*PG Scholar, Information Technology,SSEC ,Bhavnagar,Gujarat,India*
[2]*Assistant Professor Dept. of Information Technology,SSEC ,Bhavnagar,Gujarat,India*

Abstract —: *The Hadoop framework has been developed to effectively process data-intensive MapReduce applications. Hadoop users specify the application computation logic in terms of a map and a reduce function, which are often termed MapReduce applications. The Hadoop distributed file system is used to store the MapReduce application data on the Hadoop clusternodes called Datanodes, whereas Namenode is a control point for all Datanodes. While its resilience is increased, its current data-distribution methodologies are not necessarily efficient forheterogeneous distributed environments such as public clouds.This work contends that existing data distribution techniques are not necessarily suitable, since the performance of Hadoop typically degrades in heterogeneous environments whenever data distribution is not determined as per the computing capability of the nodes. The concept of data-locality and its impact onthe performance of Hadoop are key factors, since they affect the performance in the Map phase when scheduling tasks. The task scheduling techniques in Hadoop should arguably consider data locality to enhance performance. Various task scheduling techniques have been analysed to understand their data-locality awareness while scheduling applications. Other system factors also play a major role while achieving high performance in Hadoop data processing.*

Keywords- *Hadoop,Data-Locality,Distributed Computing,Big Data,Cloud Computing.*

INTRODUCTION

Meaning of the distribution means to “divide”. Here we have to perform some tasks, processes, programs, logics and the particular computations of the user.Using distributed computing we can also create remote environment in which client can access their data from different location or region from different Pcs.So finally we have to combine these different independent computers to create single system. Each node perform their tasks independently and communicate with another node using the concept of message passing. **Examples**: network of different branch office computers of one company or institute.

The current Apache Hadoop is an open-source framework used for distributed large data storage and processing of big data sets using the MapReduce programming model. Wide area network is covered by Distributed computing. Hadoop is one of the popular framework for distributed computing.A Hadoop cluster follows parallel architecture. so that store and process big data sets on the parallel machines of the hadoop cluster. Generally, the Hadoop cluster provides remote data access. The clients can use their own devices to perform the computation in Hadoop.

Figure(1) shows Hadoop architecture.The description about Hadoop architecture shown below.

Resource Manager (RM): Runs on master node**,**Global resource scheduler**,**Arbitrates system resources between competing applications

Node Manager (NM): Runs on slave node **,**Communicates with RM

Containers: Created by RM upon request, Allocate certain amount of resources (CPU cores, memory) on a slave node, Applications run in one or more containers

Application Master: One per application, Requests more containers to run application task.

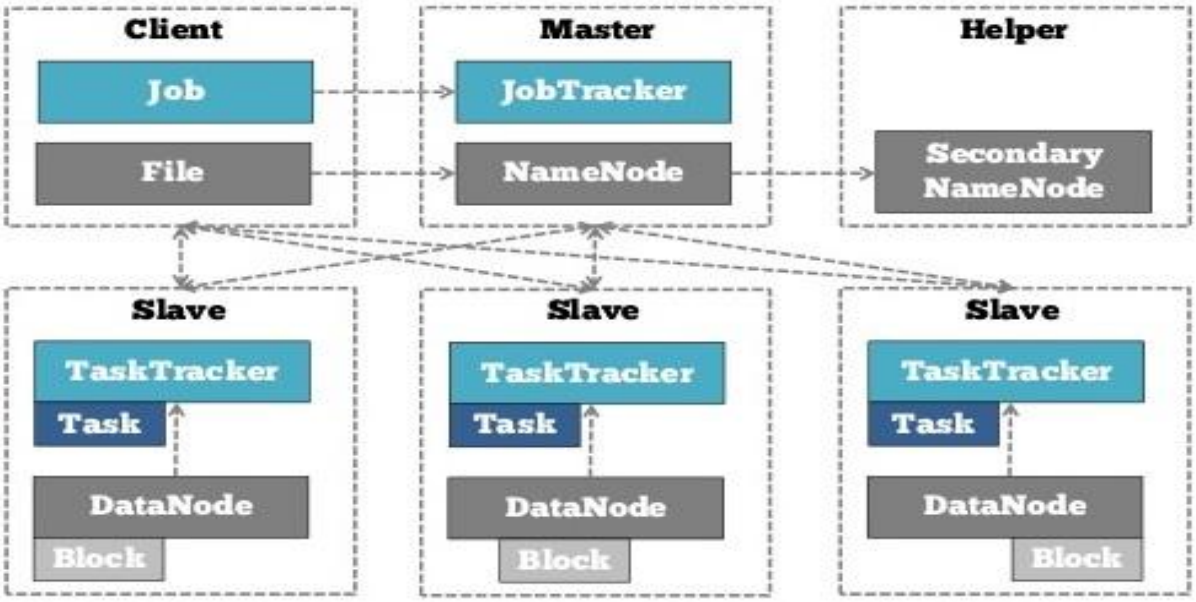

Figure 1. Hadoop Multi-Node Cluster Architecture

I. PROBLEM STATEMENT

The default Hadoop follows random and uniform data placement strategy. Hence it faces issues related to distribution and locality of data. This leads to degraded performance of Hadoop cluster. Also the techniques implemented so far doesn't take the resource parameters of the nodes in Hadoop cluster into consideration directly. They either calculate the computing capacity of nodes, computing capacity ratio of the nodes, or are based on correlation model. An optimal data placement approach is required which will ensure proper data placement based on the node's true ability to process the workloads.

The scope of this work is to study data distribution in Hadoop environment with respect to the following issues identified:

Data Distribution Issue:
Refers to the random data distribution in Hadoop cluster nodes. Poor MapReduce performance, which is ideal for homogeneous clusters in terms of computing and disk capacity.

Data Locality Issue:
Refers to the proximity of data to its processing nodes. The findings show that resources in a Hadoop cluster like CPU, memory, I/O, etc. are underutilized when the cluster is running on data-intensive applications.

II. PROPOSED WORK

Proposed system architecture is shown in figure(2).The description of that model is following:

Resource Manager
YARN Resource Manager will be responsible for managing the resources like CPU and memory. The Data Placement module will be integrated with YARN Resource Manager. The Resource Manager will keep health check of the DataNodes by pinging onto the Agent module of DataNodes after a fixed time interval.

Agent / Node Manager
The agent will respond to the ping from the Resource Manager with the health check information of the DataNodes. Also, it will forward the parameters like CPU and memory to the Resource Manager.

DataNode Profiler
This will maintain and keep the DataNode information from the Resource Manager updated. It will do DataNode profiling based on the CPU usage of the nodes.

Queue Manager
This Queue will keep the profiled DataNode information in a sorted manner. The information from this queue will be fed to the Data distribution module in the Resource Manager which will be responsible for optimal fair data distribution.

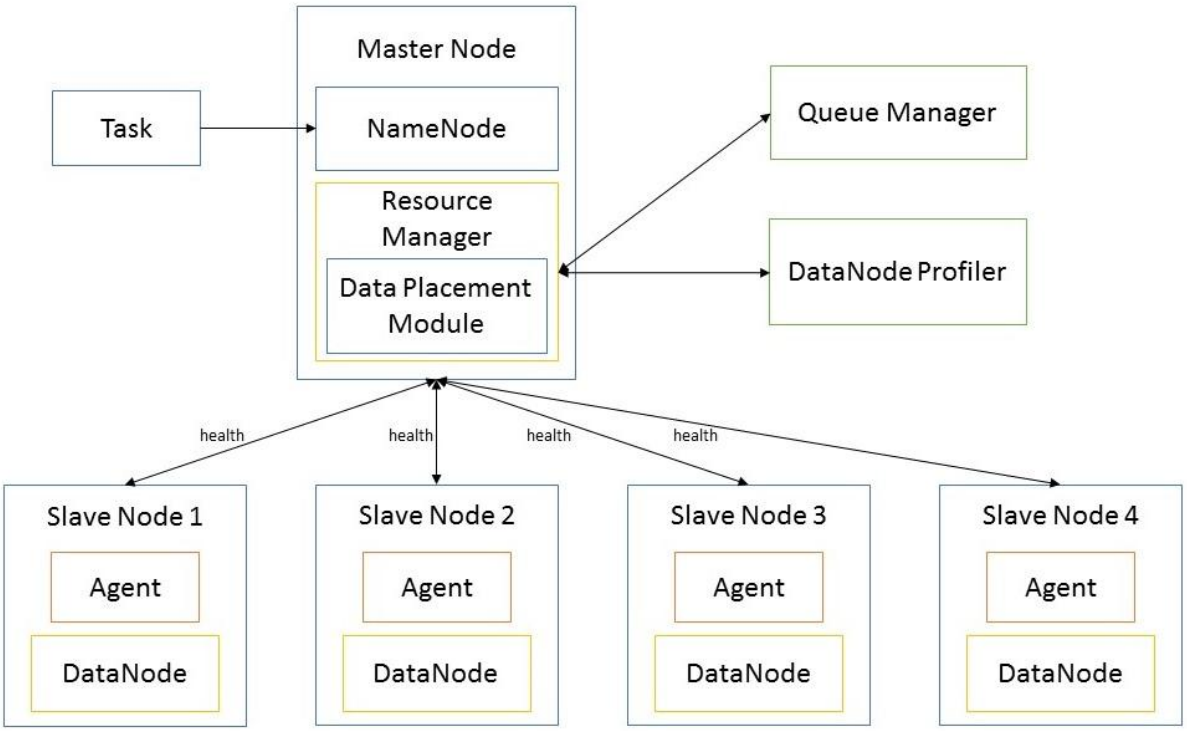

Figure 2. Proposed System Architecture

III. DISTRIBUTION STRATEGY

Phase 1: Monitoring and fetching the resource parameters of the DataNodes

- The Resource Manager will monitor the resource parameters of the Datanodes.
- Nodes profiling will be done based on the resource availability of the DataNodes.

Phase 2: Placement of data optimally after fetching the required resource parameters

- The Priority Queue module will work to sort the nodes according to their resource parameters.
- The higher the performance of node, higher will be its priority and vice-versa.
- Then onwards, whenever a new job request will arise, the data placement will be done on the higher performance nodes first.

IV. PROPOSED ALGORITHM

- ✓ *DN = Data Node; NN = Name Node; RM = Resource Manager; J = Job Request*

ALGORITHM

1. Job request J;
2. Job assigned task between NN and DN
3. For i = number of DN from 1 to n
4. RM fetches the DN resource parameters[thresold and response time]
5. Data is processed and stored in vector
6. Profile schedules over specific interval of time to fetch data
7. Queue stores the DN[i]
7. If new job request J;
8. For i = number of DN from 1 to n
9. If i == n;
10. do i = 1;
11. NN gets the D[i] for the job from the RM
12. J[i] assigned to D[i] obtained from the NN
13. DN[i] = DN[i] + 1;
14. Goto Step 7
15. Else Stop

Experimental Parameters and Evaluation Parameters

- CPU Usage (amount of CPU consumption)
- Data Size (size of input data for processing)
- **Response Time** of the application (in seconds) = Job Finish Time – Job Submit Time

V. IMPLEMENTATION ENVIRONMENT

Tools & Technologies

- Apache Hadoop 2.6.1
- Linux/Unix based, Mac, or Windows OS Platform
- Eclipse IDE
- Language: Java

Introduction about Environment

Hadoop is a well-known implementation of the MapReduce model platform, which is an open-source supported by the Apache Software Foundation. Hadoop is user-friendly for distributed appli-cation developers because it mitigates complicated managements. Hadoop consists of two main parts: MapReduce [5]and Hadoop Distributed File System (HDFS) [4], in which MapReduce is re-sponsible of parallel computing and the HDFS is responsible for data management. In the Hadoop system, MapReduce and HDFS management parallel process jobs and data, respectively. Hadoop partitions a job and data into tasks and blocks, and assigns them to nodes in a cluster. Hadoop adopts master/slave architecture, in which a master node manages other slave nodes in the clus-ter. In the MapReduce model, the master is called *JobTracker*, and each slave is called *TaskTracker*. In the HDFS, the master is called *NameNode*, and each slave is called *DataNode*. Job and data distri-butions are managed by the master to assign nodes for computing and storing. In the default setting of Hadoop, node computing ca-pacity and storage capacity are the same in the Hadoop cluster. In such a homogeneousenvironment, the data placement strategy of Hadoop can enhance the efficiency of the MapReduce model.

Hadoop uses the distributed architecture that can greatly im-prove the efficiency of reading and computing, and also uses nu-merous general PCs that can build a high-performance comput-ing platform. Spending large amounts of money to buy high-end servers is unnecessary.

For example, assume that the price of a high-end server can buy 10 or more PCs, but the performance of a high-end server is lower than 10 sets of the overall performance of the PCs. This can further reduce the cost for the data center. This is also one of the reasons why Hadoop is frequently used.

Implementation

The implementation is divided into 4 phases.

1. Create Hadoop cluster.
2. Find every node processing capacity of the Hadoop cluster to find straggler node.
3. Identification of true straggler task.(Hybrid scheduler)
4. data allocation.
5. Comparative analysis.

Step-1::Create Hadoop cluster:

The hadoop cluster is created on the amazon EC2 webservice

Node	CPU	Memory	Disk
Node A(Master)	2	2 GB	50 GB
Node B(Slave)	4	4 GB	50 GB
Node C(Slave)	2	2 GB	50 GB

Table-1 Hadoop cluster configuration

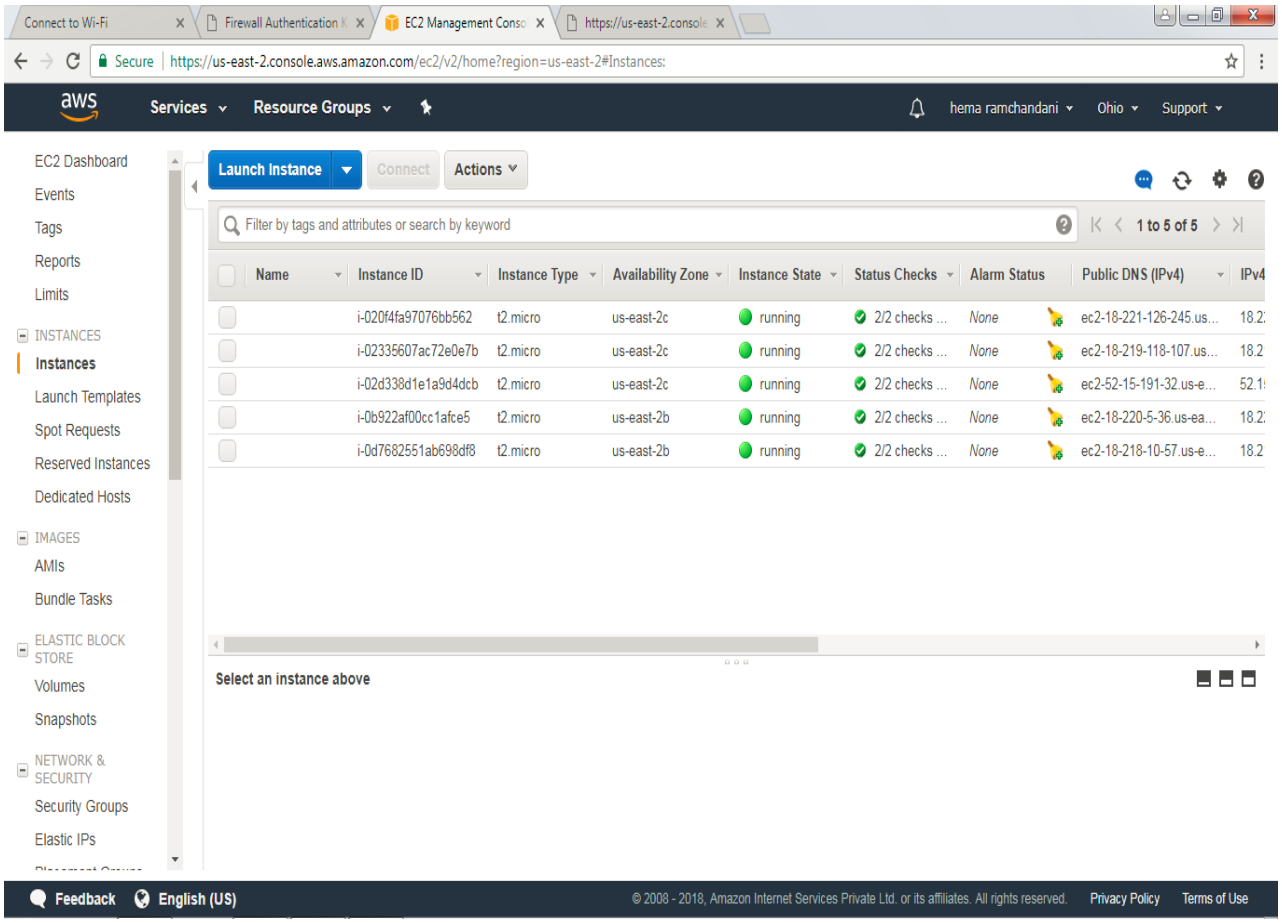

Figure 3:Hadoop cluster on aws EC2 sevice

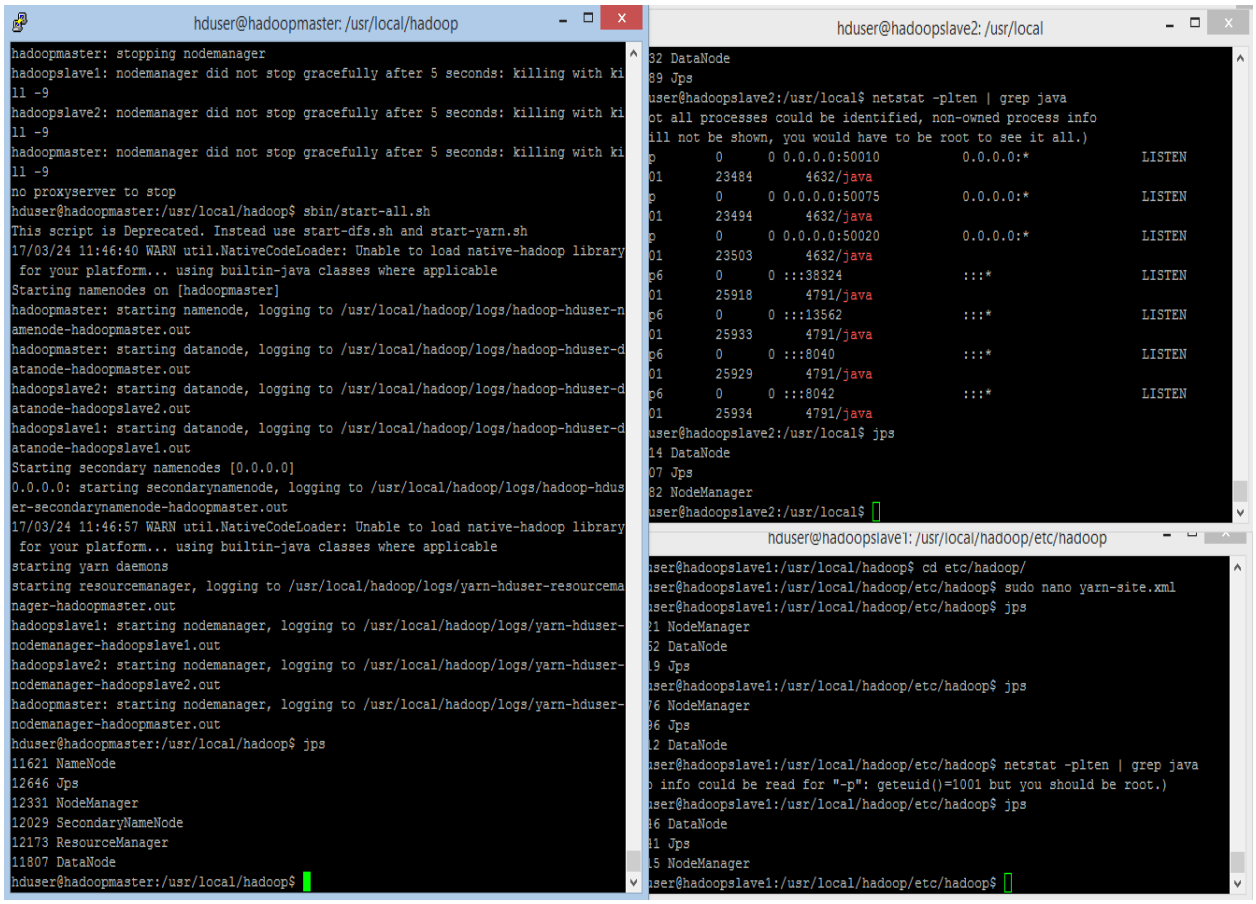

Figure 4: Daemons Running on the Master and Slave Nodes

Figure 5:Hadoop Dashboard

Step-2::Fetching parameters:

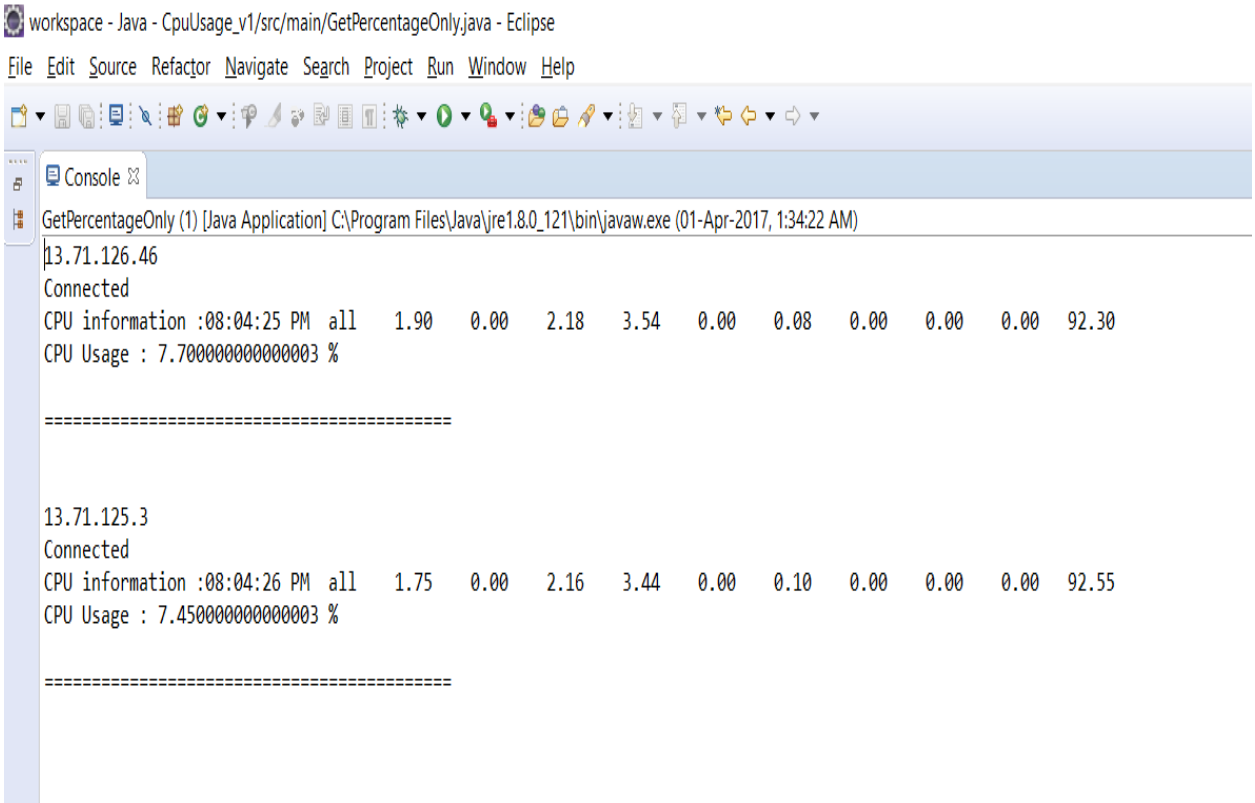

Figure 6:Fetching Parameters of cpu

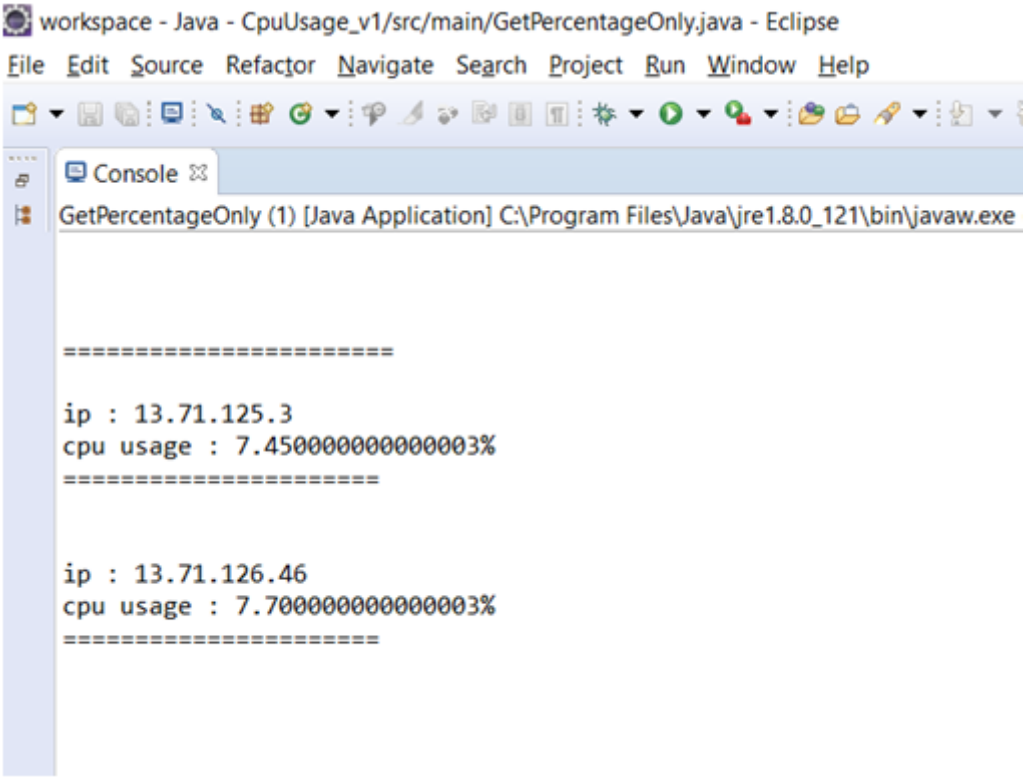

Figure 7:Sorted node according to the parameters of cpu

Step-3:: Identification of true straggler task.(Hybrid scheduler)

- If a task is not a slow task, then it performs a normal action
- If a particular task is a Straggler task, then it checks whether it is a map or reduce task.
- If it is a map task, then it checks for the availability of Map slots on the corresponding fast node.
- If Map slots are not available; then the node will wait for threshold period.
- If Map slots are present in FPN (Fast Processing Node), then schedule slow map task to the FPN.
- If slow map tasks assigned to the slow processing node; then it will receive the negative rewards.
- If slow map tasks assigned to the fast processing node,then it will receive the positive rewards.
- If it is a reduce task, then check for the Reduce slot availability on the corresponding fast node.
- if Reduce slots are not available, then the node will wait for a threshold period.
- If Reduce slots are present in FPN, then schedules slow reduce task to the fast processing node.
- If slow reduce tasks are assigned to the slow processing node; then it will receive the negative rewards.
- If slow reduce tasks are assigned to the fast processing node, then it will receive the positive rewards.
  Each TaskTracker in the cluster will update its status by sending a heartbeat message to the JobTracker.
- Then, JobTracker will update its metadata information.

Step -4:Performance Evaluation of the Proposed System

For the evaluation purpose, we ran two sample applications – WordCount MapReduce MapReducejob. The test was performed on two sizes of input file – 512 MB and 1024 MB respectively. WordCount job was performed for the same files. And then the results of the execution time taken by default Hadoop and proposed solution were compared.

The observed execution times for sample MapReduce job are as under:

Method Used	**WordCount Execution Time for 512 MB Data Size**
Default Hadoop	117 seconds
Proposed Solution	112 seconds

Table 2: WordCount job Execution Time for 512 MB Data Size

Method Used	WordCount Execution Time for 512 MB Data Size
Default Hadoop	212 seconds
Proposed Solution	205 seconds

Table 3: WordCount job Execution Time for 1024 MB Data Size

As it is evident form the table, we observed a reasonable difference in the execution times between the default approach and the proposed approach in Hadoop. X-axis refers data size and y-axis refers time in seconds.

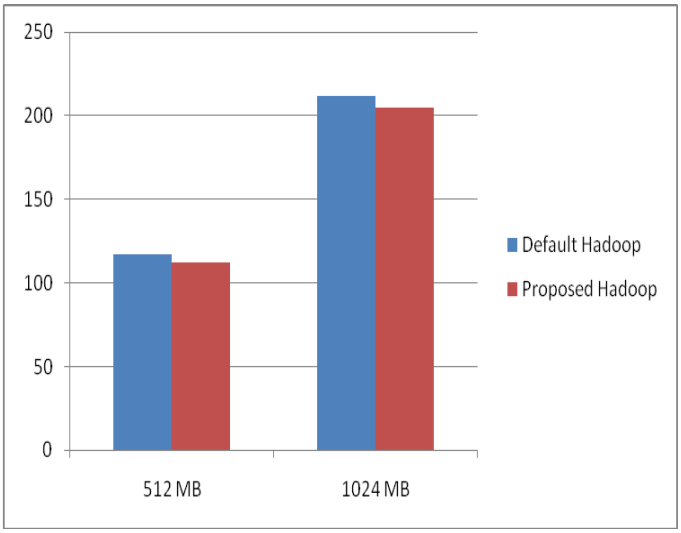

Figure 8: Execution Times for the WordCount job for Default Hadoop and the Proposed Solution

The proposed solution enhanced the response time of the job as clearly seen from the graph. It reduced the execution time by 1.71% for the 512 MB size input data and 2.359% for 1024 MB size input data. The evaluation was performed on a cluster of 4 nodes. Still it yielded notable enhancement in the response time of the jobs and improved the overall performance of the Hadoop cluster. This tends to imply that it could possibly bring considerable improvement of the Hadoop cluster by efficiently utilizing resources on large-scale Hadoop clusters. The proposed approach not just betters the performance but also resolves the straggler node issue as straggler nodes are not utilized due to their slow or poor performance.

VI. CONCLUSION AND FUTURE WORK

This dissertation work presents an optimal data placement strategy in Hadoop environment. It describes a proper data placement design for Hadoop clusters. This dissertation explores the Hadoop data placement policy in detail and proposes a modified data placement approach that increases the performance of the overall system. The idea of placing data across the cluster according to the utilization of the nodes resources is presented, which will enhance the job execution/response times and improving the overall Hadoop cluster performance by efficient utilization of resources. The proposed solution fairly considers each node's resources and distributes the data based on their utilization. The new data placement strategy could bring reasonable improvements in large-scale Hadoop clusters. In future, the proposed work remains to be tested over a larger number of nodes to find out the true potential of the new system.

REFERENCES

[1] Suhas V. Ambade, Prof. Priya R. Deshpande, "Heterogeneity-based files placement in Hadoop cluster",2015 IEEE, Pages: 876 - 880, DOI: 10.1109/CICN.2015.325

[2] C.-W. Lee et al., A Dynamic Data Placement Strategy for Hadoop in Heterogeneous Environments, Big Data Research (2014), http://dx.doi.org/10.1016/j.bdr.2014.07.002

[3] Jia-xuan Wu, Chang-sheng Zhang, Bin Zhang, Peng Wang, A New Data-Grouping-Aware Dynamic Data Placement Method that Take into Account Jobs Execute Frequency for Hadoop, Microprocessors and Microsystems (2016), DOI: 10.1016/j.micpro.2016.07.011.

[4] B Ganesh Babu, Shabeera T P, Madhu Kumar S D, "Dynamic Colocation Algorithm for Hadoop", 2014 IEEE, Pages: 2643 – 2647, DOI: 10.1109/ICACCI.2014.6968384

[5] Vrushali Ubarhande, Alina-Madalina Popescu, Horacio Gonzalez-Velez, "Novel Data-Distribution Technique for Hadoop in Heterogeneous Cloud Environments", 2015 IEEE, Pages: 217 - 224 DOI 10.1109/CISIS.2015.37

[6] Jiong Xie, Shu Yin, Xiaojun Ruan, Zhiyang Ding, Yun Tian, James Majors, Adam Manzanares, and Xiao Qin, "Improving MapReduce Performance through Data Placement in Heterogeneous Hadoop Clusters", 2010 IEEE, Pages: 1 – 9, DOI: 10.1109/IPDPSW.2010.5470880

[7] Yuanquan Fan, Weiguo Wu, Haijun Cao, Huo Zhu, Xu Zhao, Wei Wei, "A heterogeneity-aware data distribution and rebalance method in Hadoop cluster", 2012 IEEE, Pages: 176 – 181, DOI 10.1109/ChinaGrid.2012.22

[8] Stephen Kaisler, Frank Armour, J. Alberto Espinosa, William Money, "Big Data: Issues and Challenges Moving Forward", 2012 IEEE, Pages: 995 – 1004, DOI 10.1109/HICSS.2013.645

[9] Jun Wang, Qiangju Xiao, Jiangling Yin, and Pengju Shang, "DRAW: A New Data-gRouping-AWare Data Placement Scheme for Data Intensive Applications with Interest Locality", 2013 IEEE, Pages: 2514 – 2520, DOI: 10.1109/TMAG.2013.2251613

[10] Hellerstein, Joe (9 November 2008). "Parallel Programming in the Age of Big Data". Gigaom Blog

[11] *Segaran, Toby; Hammerbacher, Jeff (2009).* Beautiful Data: The Stories Behind Elegant Data Solutions. *O'Reilly Media. p. 257.* ISBN 978-0-596-15711-1

[12] The Apache Software Foundation. Hadoop. http://hadoop.apache.org

[13] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. Commun. ACM, 51(1):107–113, 2008.

[14] New Horizons for a Data-Driven Economy – Springer, doi:10.1007/978-3-319-21569-3.

[15] The World's Technological Capacity to Store, Communicate, and Compute Information". MartinHilbert.net.

[16] JASON. 2008. "Data Analysis Challenges", The Mitre Corporation, McLean, VA, JSR-08-142

[17] "Welcome to Apache Hadoop!". hadoop.apache.org

[18] "What is the Hadoop Distributed File System (HDFS)?". ibm.com. IBM

[19] Malak, Michael (2014-09-19). "Data Locality: HPC vs. Hadoop vs. Spark". datascienceassn.org. Data Science Association

