International Journal of New Technology and Research (IJNTR)
ISSN:2454-4116, Volume-3, Issue-4, April 2017 Pages 73-76

An Automatic Alignment and Grouping of Hadoop Cluster

Dr V. Govindasamy,P. Siva Kumar, K. Puviarasu, K. Vijiya Kumar

Abstract— **The MapReduce Framework and Hadoop is the platform for scalable analysis on large Datasets in recent years. The primary concern in the Hadoop is to minimize the completion length (i.e., makespan) and fixed number of MapReduce jobs. This makes performance low and causes low resource utilization. To overcome this, we propose a system which dynamically allocates the map and reduce jobs, thus leading to high resource utilization and reduced completion length. The dynamic allocation of MapReduce jobs is achieved by implementing Combiner Interface in MapReduce Framework. The Proposed solution is implemented in the Amazon EC2 Cluster in both Homogeneous and Heterogeneous Clusters. The experimental results show the effectiveness and robustness of our proposed system under simple workloads.**

Index Terms— **Map Reduce Slots, Hadoop Scheduling, Hadoop Cluster, Reduced Makespan.**

I. Introduction

In today's world, ahuge amount of data is generated, it is somewhere eating the storage space. At one side, the technology gears up to provide more space and cloud technologies are the requirement of storage technologies for anenormous amount of data. The Huge amount of data is generated and most of the companies analyze them and make a huge amount of profits from that data. In a minute, $400 million Dollar sold on Alibaba, 4,39,000 page views on Wikipedia, 1, 94,000 applications downloaded from the play store, 38, 000 photos uploaded on the Instagram, 10 Million advertisement's, 4.1 million searches on Google around the globe. Thus, in one year, the amount of data generated is unimaginable. To analyze this much of huge amount of data and store them we need special storage techniques and processing techniques.Here comes to the picture, Hadoop [3]. It is the tool which is used to store huge amount of data and analyze them.

Hadoop [3] is the open source software framework for distributed storage and parallel processing of the huge data sets through programming model MapReduce [2]. The Hadoop consists of a storage part known as Hadoop Distributed File System (HDFS) [2] and processing part which is MapReduce programming model. The classic Hadoop cluster relies on the Master- Slave Architecture, In Hadoop cluster single Master node and multiple Slave nodes. The Master node is known as the Namenode which is responsible for the job allocation in the slave nodes. The slave nodes in Hadoop cluster known as the DataNode which is controlled by the NameNode. Hadoop splits the larger data set files and distribute across the nodes in the cluster, then transfer the code to thenodes to process the data in parallel[3]. This approach takes advantage of the parallel processing, this allows data set process faster and more efficiently that relies on the parallel file system. The Master node runs JobTracker, which is responsible for scheduling the jobs and coordinating the execution of each job. Each Slave node runs the TaskTracker for the execution of MapReduce jobs. Hadoop is a fault tolerant which is protected against the hardware failure,if a node goes down, jobs are automatically redirected to other nodes to make sure distributed system does not fail, multiple copies of data are stored automatically in thecluster.

Dr V. Govindasamy,Associate Professor, Department of Information Technology, Pondicherry Engineering College.
P. Siva Kumar, Student, Department of Information Technology, Pondicherry Engineering College.
K. Puviarasu, Student, Department of Information Technology, Pondicherry Engineering College.
K. Vijiya Kumar, PhD, Department of Computer Science, Pondicherry Engineering College.

A. Scope:

The performance can be improved by the Dynamic allocation of the MapReduce jobs [1], the transfer of data between the map and reduce jobs is directly implied on the performance of Hadoop cluster. If thetransfer of data between the Map and Reduce job is minimal means resources in the cluster will be more and automatically performance will be improved.

In this work, we propose the Dynamic allocation of MapReduce jobs and Improvement in makespan (i.e., completion length). The key idea of the new mechanism is implementing a Combiner interface in the MapReduce, which will decrease the transfer of data between Map and Reduce. The Dynamic allocation of Map and Reduce jobs is achieved by the implementing the MapReduce framework in YARN, this system provides the dynamic allocation of MapReduce jobs.

The rest of the paper is organised as follows, we explain the YARN in section II. The design of the proposed solution is explained in section III. The environment set-up is explained in section IV and experimental results and graphs are formulated in section V. Section VIconcludes the future work. We conclude in Section VII.

II. YARN

In Hadoop 2.0 the new component is added that is called the YARN (Yet Another Resource Negotiator) [3]. In YARN, Cluster management is performed and data processing is performed by the MapReduce. The architecture of the Hadoop 2.0 is specified in Fig 1.

A. *Resource Manager:*

It is a Master and Global Resource Manager. The Resource Manager has a scheduler, which is responsible for allocating resources to the various applications running in the cluster, according to constraints such as queue capacities and user limits. The scheduler schedules based on the resource requirements of each application.

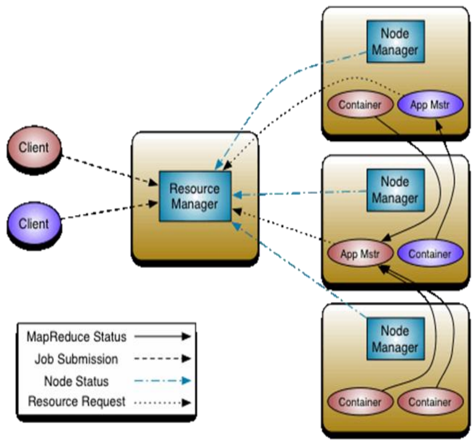

Fig 1: YARN Architecture

B. *Node Manager:*

It is in the slave state. It is responsible for the Resource Management on Data node and it is also responsible for the both Application Manager and Container. It is also responsible for the monitoring their resource usage (CPU, memory, disk, network) and reporting the same to the Resource Manager.

C. *Application Manager:*

Application Manager is responsible for the job execution and it can run on any Data node. It is available in the Node Manager and it will die automatically after job dies. It is also responsible for allocating the Map and reducing tasks to execute the job. Each Application Manager has responsibility for negotiating appropriate resource containers from the scheduler, tracking their status, and monitoring their progress. From the system perspective, the Application Manager runs as a normal container.

D. *Container:*

It is created in the Node Manager and also it takes the details from the Resource Manager. In simple terms, Container is a place where a YARN application is run. It is available in each node. Application Manager negotiates container with the scheduler (one of the components of Resource Manager). Containers are launched by Node Manager.

III. SYSTEM ARCHITECTURE

When dealing with the Big Data analysis the data should be very accurate without any abnormalities. There are numerous factors that affect the performance of the Hadoop cluster, such as the both Hardware and Software while handling the huge amount of the data. Both the main components of the Hadoop i.e., HDFS [3] and MapReduce [2] plays a major role in the performance of the cluster. By using these Diagnostics [6], we can improve the performance in the cluster:

i) ConFig your cluster correctly
ii) Tune the number of maps and reduce tasks appropriately
iii) Write a Combiner
iv) Use the most appropriate and compact Writable type for data set

For analyzing the enhancement performance in the Hadoop cluster, this can be achieved by the enhancing the MapReduce jobs. We need:-

i) Dataset: In order to evaluate the performance we need the dataset which is big or huge data set through which we can calculate the performance.

ii) Hadoop: Hadoop should be conFigd first because all the MapReduce job will work on Hadoop framework because Hadoop comes with HDFS(Hadoop Distributed File System) which is used to stored such huge or large datasets and MapReduce which is used to process this huge dataset.

iii) MapReduce jobs: MapReduce jobs can be developed on any IDE through which we can develop the various MapReduce jar file which willbe useful for the performance evaluation.

In this work, we propose the new mechanism to dynamically allocate the map and reduce slots. The main aim of this mechanism is to improve the completion time of a batch of map reduce in the Hadoop cluster. The key idea of the new mechanism is the Combiner. A combiner is also known as the semi reducer which accepts the inputs from the map class and there passes them to the reducer class. The combiner is used between the map class and the reducer class to reduce the transfer of data between the map classes and reduce class. The Architecture overview of proposed system is shown in Fig 2.

A. *Data Cleaning:*

In proposed system, first, the user should upload the Input Data Set and then Data set should be cleaned which means that the data cleaning should be done. In Data cleaning process the noisy data should be cleaned in the data set, the unnecessary Data values are to be removed which are not useful for the Analysis. Then pre-processed Data set is should be sent to the next stage.

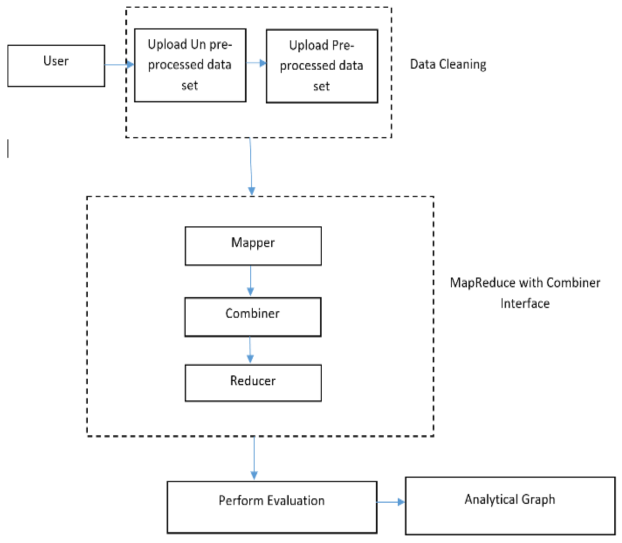
User
Upload Un pre-processed data set
Upload Pre-processed data set
Data Cleaning
Mapper
Combiner
Reducer
MapReduce with Combiner Interface
Perform Evaluation
Analytical Graph

Fig 2: Overall System Architecture

B. *MapReduce with Combiner Interface:*

A Combiner [4], is an optional class that operates by accepting the inputs from the Map class and passes the output key-value pairs to the Reducer class. The main function of a Combiner is to summarize the map output records with the same key-value, the output key-value collection of the combiner is sent over the network to the actual Reducer task as input. A combiner is also known as the Semi-reducer.

The data transfer between Mapper and Reducer can be minimized by the implementing the Combiner in between Mapper and Reducer. Generally, the output of the map task is large and the data transferred to the reduce task is high. By implementing the Combiner in the MapReduce framework it offers a network congestion.

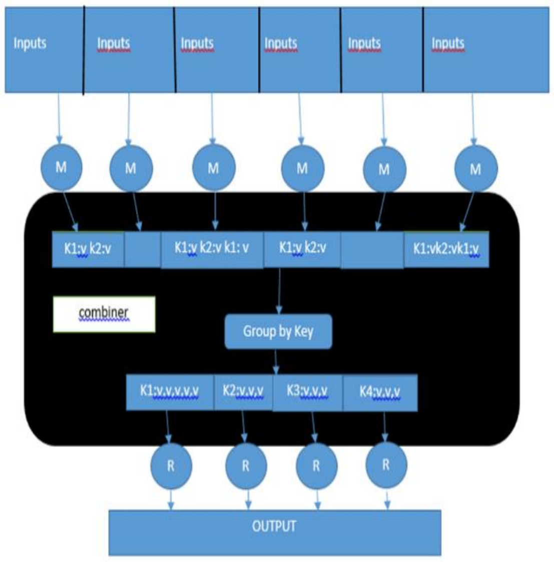

Fig 3: Map Reduce with Combiner

Working of the Combiner:

The working of the combiner is explained below:

1. A combiner does not have any predefined interface and it must be implemented by the Reducer interface's reduce ( ) method.

2. A combiner operates on each and every map output key. It must have the same output key-value types as the Reducer class.

3. A combiner can produce summary information from a large dataset because it replaces the original Map output.

C. *Performance Evaluation:*

The Performance Evaluation of the proposed system is compared with the Fair scheduler of the Hadoop Scheduler and then performance is evaluated in Hadoop Cluster in both the Homogeneous and Heterogeneous Environments. The comparison is done between the MapReduce and MapReduce with Combiner Interface.

D. *Analytical Graph:*

The Results are obtained from the implementation which is represented in Graph for easy understanding and analysis of the system.

IV. ENVIRONMENT SET-UP

In environment setup we created the cluster by using the CentOs, by using this we created the Master node (Name node) and Slave node (Data node). In our problem we created the one Master node and three slave nodes. The different types of the Hadoop cluster is

i) Homogeneous Cluster:

In a Homogeneous Hadoop cluster [1] environment, where all servers have the same computing and memory capacities. In this environment, tasks from the same jobs will have the same performance/execution time.

ii) Heterogeneous Cluster:

In a Heterogeneous Hadoop cluster [1] environment, where all servers does not have the same computing and memory capacities. In this environment, tasks from the same job may have the different performance /execution time when running on the different nodes. In this case task execution time highly depends on a particular node where the task is running. A job's map task may run faster on a node which has faster CPU per slot while it reduce tasks may experience shorter execution time on the other nodes that have more memory per slot. After the creation of the Cluster we will initialize the Hadoop Cluster components like Name Node, Data Node, YARN Components and HDFS.

V. RESULTS

In this work, we tested the various size of the Data sets [5] for our proposed solution and we shows the comparison between the existing work and proposed solution. The performance improvement between Hadoop fair scheduler and the proposed solution is shown in table and plotted in the graph.

Table 1: Execution time of Sample Data Set

Execution time	Milliseconds
With combiner	3837
Without combiner	111042

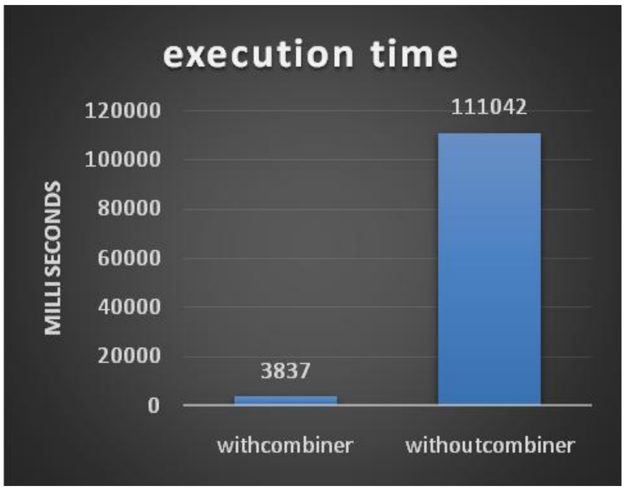

Fig 4: Execution time of Sample DataSet

We tested our Proposed System, in different data set sizes which shows the performance variation, which states that i.e.,

Table 2: Execution time of Different Sizes of Data Sets

	Milliseconds	
Input size	Combiner	Without combiner
1 GB	296278	420268
500 MB	115016	185020

if performance increases means automatically resource utilization will also be high. In Fig 5 we shown the Performance variation on our test cases with different input sizes which are stated in the below table 2.

In Fig 4, shown the performance difference on the sample data set with our proposed solution and existing system. In table 2 we stated that the different sizes of data sets used for test analysis for our proposed solution. For our test analysis we implemented graph for easy analysis which is stated in Fig 5.

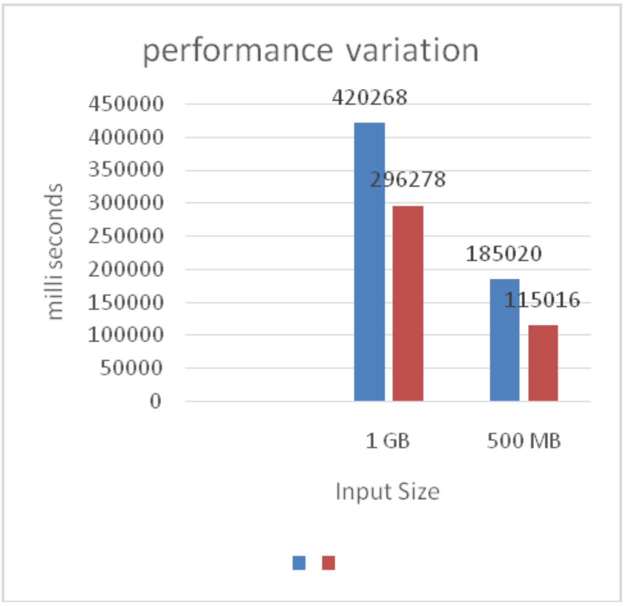

Fig 5: Performance Variation

VI. FUTURE ENHANCEMENTS

In future, we will implement the proposed system in the complex workload and we can also further investigate in the machine learning algorithm to find the fault nodes in the cluster and reconstruction of the node in the cluster.

VII. CONCLUSION

The static allocation of Map Reduce jobs in the cluster which directly implies on the performance time of the cluster. We presented a dynamic allocation of Map Reduce slots in the Hadoop cluster in heterogeneous environment, implementing the Combiner Interface. The Combiner Interface will reduce the transfer of the data between the Map Reduce Interface. The transfer of data is less means resource availability will be more, performance will be improved automatically.

REFERENCES

[1] Yi. Yao, Jiayin Wang, Bo Sheng, Chiu C. Tan and Mi, “Self –Adjusting Slot Configuration for Homogeneous and Heterogeneous Hadoop Clusters”, IEEE Transactions on Cloud Computing, pp. 99-105, 2015.
[2] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters”, Communicatio of the ACM, vol. 51, no. 1, pp. 107-113, 2008.
[3] http://hadoop.apache.org/
[4] Prajesh P Anchalia, “Improved MapReduce k-Means Clustering Algorithm with Combiner”, UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, 2014.
[5] https://aws.amazon.com/datasets/
[6] http://blog.cloudera.com/blog/2009/12/7-tips-for-improving-mapreduce-performance/