ISSN:2320-0790

An International Journal of Advanced Computer Technology

Deploying and Improving Hadoop on Pseudo-Distributed Mode

Raj Kumari Bhatia, Aakriti Bansal
University Institute of Engineering & Technology
Panjab University, Chandigarh
India

***Abstract*: Hadoop is an open source framework which comprises of MapReduce and HDFS (Hadoop Distributed File System) which allows storage of data and computing capabilities on large scale. Hadoop is expanding every day and many cloud computing enterprises have been adapting it. Hadoop provides a platform to provide cloud computing services to customers. Hadoop can be run of any the three modes: Standalone, Pseudo-Distributed and Fully-Distributed. In this paper we have improved execution time by configuring different schedulers. We implemented our method over Hadoop 2.2.0 on Pseudo-Distributed mode which improves the overall job execution time.**

Keywords- Hadoop, MapReduce, Schedulers

I. INTRODUCTION

In today's world billions of people are using Internet and with usage of internet at this scale amount of data that flows over the internet is humongous and with cloud computing this data is increasing even at a higher rate, [1]. This amount of data which is transferred daily from one location to another over the internet has made computing a very complicated task. To manage such huge amount of data is in itself a challenging task. Complex algorithms need to be written to accomplish this task but Hadoop provides us with a good option to reduce the complexity of managing this data.

Managed by Apache Software Foundation, [2], Hadoop is open source software which implements MapReduce algorithm used by Google to manage data. MapReduce framework operates on <key, value> pairs, in which I/O is both a set of <key, value> pairs.Hadoop, today is one of the most trending technology which has gained attention of many big enterprises and individuals. Hadoop solves problem of managing too much data which is mixture of structured and complex data which does not fit into tables.

Hadoop can be used in various markets like Finance, Online Retail, Portfolio evaluation, Risk Analysis and many more.

Hadoop runs on large number of machines which do not share memory or disks. A Hadoop cluster consists of servers which consist of two to eight CPUs, each CPU consisting one big disk connected to 4-16 big processors. Hadoop spreads out silos of data to deal with it and all these big processors work in parallel to answer complicated questions.Processors working in parallel utilize CPU resources more effectively. Hadoop provides three Job Schedulers which are: Job Queue Task Scheduler, Fair Scheduler and Capacity Task Scheduler which are described later in the paper. Execution time of a job has always been a concern in Hadoop framework as one slow job can affect the overall execution time.

Hadoop can be run in three different modes: First is standalone mode which is default mode of Hadoop and HDFS is not used in this mode. Second is Pseudo-Distributed mode which is a single node cluster and uses HDFS as file system. Third mode is Fully-Distributed cluster which has multiple nodes and data is used and processed among these nodes.

Master Nodes, Slave Nodes, Data Nodes and Name Nodes are used as separate entities.
In this paper we propose a way to reduce the execution time of a single job on Hadoop.
The rest of the paper is organized as follows. The Research Background in section 2 and Implementation is written in section 3. The Results in section 4 and Limitations in section 5. Finally we highlight the Conclusion and Future work at the end of the paper in section 6.

II. RESEARCH BACKGROUND

A. Map Reduce

The MapReduce performs two distinct phases in Hadoop. First is Map phase which takes set of data and process itin parallel to returna set of data as intermediate result and another is Reduce phase which takes intermediate results and reduces it in smaller set of data. Overall MapReduce condenses large amount of data in useful data. It divides a query into multiple steps.
MapReduce has emerged from an interesting paper by Google in 2004 to an industry standard. It can be used to filter documents by tags, count number of words, artificial intelligence, large scale PDF generation, to process geographical data and many more tasks. It should not be used when latency needs to be predictable.

B. Hadoop

Hadoop is an open source Java based framework to store and process large amounts of data. It allows distributed processing of data which is present over clusters using functional programming model. The storage component of Hadoop is called HDFS and processing component of Hadoop is called MapReduce.
Hadoop Distributed File System (HDFS) is java based file system which provides reliable and robust data storage. HDFS consists of NameNodes which manage metadata and DataNodes where file data is stored. Metadata is represented on NameNodes as Inodes which record information like permissions, access times etc. NameNodes send instructions to DataNodes which include commands according to [3].
MapReduce is the most important algorithm implemented in Hadoop. Each Map and Reduce is independent of other Maps and Reduces. Processing of data is executed in parallel to other processes. A job scheduler or job tracker tracks MapReduce jobs which are being executed. Tasks like Map, Reduce and Shuffle are accepted from Job Tracker by a node called Task Tracker. Job Tracker is the master node to all slave Task Tracker nodes which use resources provided by Job Tracker to perform one or multiple tasks.

C. Advantages and Disadvantages of Hadoop 2.2.0

1) Advantages

Hadoop is highly stable and provides distributed computing and storage capabilities. Enterprise Readiness of Hadoop makes it easier to work upon it by many cloud computing enterprises. Tasks are independent of each other. Hadoop's extreme scalability, availability and fault tolerance is because of replication of data which is called replication factor by default its value is 3.According to [4], Distributed computing in Hadoop allows parallel working and gives high throughput.

2) Disadvantages

One of the few disadvantages of Hadoop is its poor performance which needs to be resolved. Another issue is cluster management.

III. IMPLEMENTATION

A. Hadoop Deployment

[5], they deployed Hadoop in virtual machines and developed a concept to decide virtual machine among many virtual machines on single physical machine. In this paper, we deploy hadoop 2.2.0 over Ubuntu 12.04 with Linux kernel version 3.8.0-29-generic. Basically hadoop is written in java. To deploy the above, we follow these steps:-

1. We Install JDK version 7 on Ubuntu using the below Linux Command.
   ```
   $sudo apt-get install openjdk-7-jdk
   ```
2. Untar hadoop-2.2.0.tar.gz file in directory location /usr/local and install it using sudo command.
   ```
   $sudo tar hadoop-2.2.0.tar.gz –C /usr/local
   ```
3. Setting Hadoop Environment Variables:
   Set environment variable JAVA_HOME to path of JDK and HADOOP_INSTALL to

path of Hadoop and PATH to sbin in Hadoop folder and HADOOP_MAPRED_HOME,HADOOP_HDFS_HOME, YARN_HOME to HADOOP

4. Now configure yarn-site.xml,core-site.xml,mapred-site.xml and hdfs-site.xml accordingly.

B. Run Hadoop

First format NameNode, which is created by previous execution of hadoop by
$hdfsnamenode –format
Then run shell files of YARN and DFS which will start Resource Manager, Node Manager, DataNodes, NameNodes and Secondary NameNodes.
After performing above steps Hadoop services are now in running state.

C. Schedulers

Three types of schedulers can be used in Hadoop: - Fair Scheduler, Capacity Scheduler, Job Queue Task scheduler.

1) Fair Scheduler

Fair Scheduler is configured at two places: mapred-site.xml and fairscheduler.xml. It is an allocation file which is loaded periodically at run time that allows changing settings without restart the Hadoop cluster. Fair Scheduler as name suggests, is a method to assign resources in such a way that on an average all jobs get equal amount of resources.
To use fair scheduler, the following property need to be set in yarn-site.xml like below:

```
<property>
<name>yarn.resourcemanager.scheduler.class</name>
<value>org.apache.hadoop.yarn.server.resourcemanager.
scheduler.fair.FairScheduler</value>
</property>
```

2) Capacity Scheduler

It allows sharing a large Hadoop clusters among many organizations and ensures a minimum capacity of resources to each organization. The resources which are not used by any organization are free to be used by any one of them, hence providing elasticity, security, operability across multiple organizations in a cost effective manner.
To use Capacity Scheduler, the following property needs to be set in yarn-site.xml like below:

```
<property>
<name>yarn.resourcemanager.schedular.class</name>
<value>org.apache.hadoop.yarn.server.resourcemanager.
schedular.capacity.CapacitySchedular</value>
</property>
```

Multiple queues can be defined using mapred.queue.names in hadoop-site.xml. To control the behavior of scheduler multiple queues can be configured with several properties in capacity-scheduler.xml.

3) Job Queue Task Scheduler

Job Queue Task Scheduler is the default scheduler provided by Hadoop and uses First In First Out (FIFO) methodology to maintain jobs.
[6], they used it to maintain the list of jobs in accordance of their submission times. Jobs run in a queue and wait for their turn until the current job has been completed with execution.

When to use each scheduler:
When running on large Hadoop Cluster which has multiple clients in different priority of jobs then capacity scheduler is recommended. Fair Scheduler is used when both small and large cluster are used by small organization with limited work load.

4) Proposed Work

We implemented our method on Hadoop 2.2.0 as follows:
At first we configured *yarn-site.xml*file in Hadoop and added property values for Fair Scheduler. By adding properties for Fair Scheduler we are pointing to Fair Scheduler algorithm, the properties of which are configured in *fair-scheduler.xml*.
In *fair-scheduler.xml* we configured allocation queues and in those queues we have set values of minimum resources and maximum resources that will be made available to a particular job. We also set values for maximum running applications, weight and scheduling policy to be followed, fair share preemption timeout property and access control list.

IV. RESULTS

We have implemented our method on Hadoop 2.2.0 and to reach to the conclusion we have taken the mean of the execution time values of a particular job for a set of map slots.
For comparison first we observed values of default Hadoop for a predefined set of number of map slots

and took mean over the set of values found for a particular number of map slots and for the same set of map slots we have observed the execution result values and took mean over the set of values found.

In all when we compared the values of default hadoop and proposed hadoop for a particular number of map slots we found that there was some significant improvement in the execution time and hence overall performance of Hadoop is improved. We have calculated our results on the below mentioned performance environment.

Version	Hadoop 2.2.0
File System	HDFS
Benchmark Program	Sort
Scheduler	Fair Scheduler
CPU	Core i3/2.13 GHz
OS	Ubuntu 12.04
Kernel	Linux 3.8.0-29
Memory Size	2 GB

TABLE 1
PERFORMANCE ENVIRONMENT

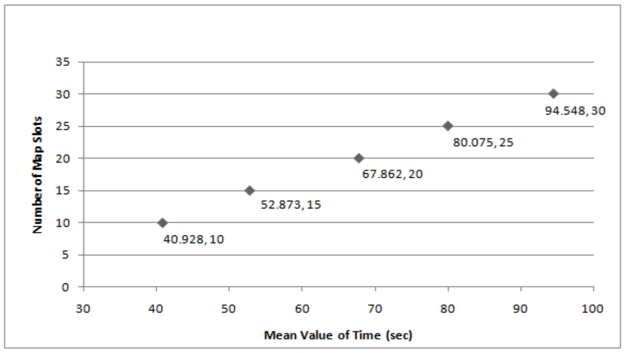

Fig.1. Execution time taken by a job with respect to number of map slots by Proposed Hadoop

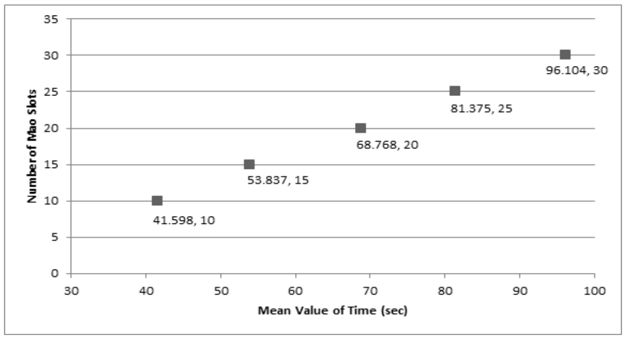

Fig.2. Execution time taken by a job with respect to number of map slots by Default Hadoop

Figure 1 and Figure 2 clearly demonstrates the difference in mean execution times for a particular number of map slots between Default Hadoop and Proposed Hadoop.

V. Limitations

As hadoop is used for large amounts of data which is generally in terabytes and petabytes and this system is tested on small amount of data.

System specifications are comparatively low to which is required for real time Hadoop execution.

Results are based on average of values that were found in multiple executions.

VI. Conclusion and Future Work

In this paper we talked about MapReduce and Hadoop. Mapreduce performs job in two phases map and reduce in functional programming language which can be optimized in parallel processing. Then we discussed about Hadoop Deployment and Running on Ubuntu 12.04 with linux kernel version 3.8.0-29-generic.We implemented the proposed method on hadoop 2.2.0 and evaluated it on terasort as benchmark programs by executing jobs. Using Modified Hadoop Job finished time is found to be improved.

Following issues can be addressed for future considerations: One is real time performance evaluation by passing large amount of data as input and getting exact output. The above proposed systemcan be implemented on Hadoop Cluster for future works.

REFERENCES

1. *Hadoop*, The Apache Software Foundation, May 2012, 1.0.3.
2. Thusoo, Ashish, et al. "Hive: a warehousing solution over a map-reduce framework." *Proceedings of the VLDB Endowment* 2.2 (2009): 1626-1629.
3. Shvachko, Konstantin, et al. "The hadoop distributed file system." *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*. IEEE, 2010.
4. Nicolae, Bogdan, et al. "BlobSeer: Bringing high throughput under heavy concurrency to Hadoop Map-Reduce applications." *Parallel & Distributed Processing (IPDPS), 2010 IEEE International Symposium on*. IEEE, 2010.
5. Xu, Guanghui, Feng Xu, and Hongxu Ma. "Deploying and researching Hadoop in virtual machines." *Automation and Logistics (ICAL), 2012 IEEE International Conference on*. IEEE, 2012.
6. Kurazumi, Shiori, et al. "Dynamic Processing Slots Scheduling for I/O Intensive Jobs of Hadoop MapReduce." *ICNC*. 2012.