A Study on Hadoop MapReduce Techniques and Applications on Grid Computing

Ila Savant
M.Tech CSE*
IES, Bhopal

Richa Muke
B.E Computer*
MMCOE, Pune

Nilay Narlawar
B.E Computer*
MMCOE, Pune

Abstract

Hadoop is an open source programming framework which provides processing of voluminous amounts of data using distributed computing environment. Hadoop creates clusters of inexpensive computers and distributes work among them. MapReduce is the core of Hadoop technology. It enables the massive scalability across hundreds to thousands of servers in a Hadoop cluster. In this paper, we focused Hadoop's MapReduce techniques and their comparative study. Also we discussed MapReduce application on grid computing, image processing to deal with big data problem.

Keywords

Distributed computing, clusters, MapReduce, Grid computing.

1. Introduction

Almost 90% of the data produced worldwide has been created in the last few years alone, that is, 2.5 quintillion bytes per day. Recent times have seen enormous volumes of data growing at a rapid rate. Therefore, new strategies and processes are needed to process this burgeoning data. In addition, a change has been observed in the format of data produced, from conventional text based data to unstructured sources including web server logs, blogs, images and social media streams. These changing data formats have overwhelmed the existing systems. Also, the old systems had some limitations. Businesses now needed not just queries but also processing. Apache Hadoop, an open source software framework, is an optimal solution to all these problems. Hadoop uses two core functionalities, namely, HDFS (Hadoop Distributed File System) and MapReduce.

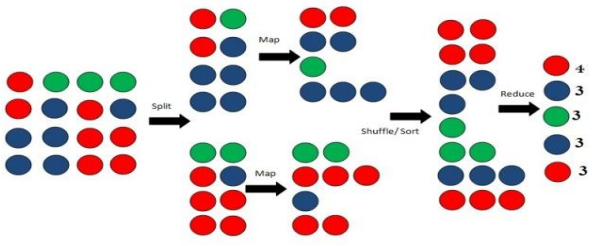

Fig.1 working of map and reduce technique

Fig.1: The input to the MapReduce process in figure is a set of colored circles. The objective is to count the number of each color. The programmers in this example are responsible for coding the map and reduce programs; the remainder of the processing is handled by the software system implementing the MapReduce programming model.

MapReduce technology was pioneered by Google for processing of their large data generated every day. It is a form of abstraction which highlights the necessary computations and hides the unnecessary and tedious details such as parallelization and load balancing. It is an implementation of the Map and Reduce functionalities used in common functional languages. Map will map the records in input, while Reduce will combine all the values having same key. This allows parallelization to a large extent.

Hadoop includes a huge storage system called the Hadoop Distributed File System (HDFS). It is capable of storing massive amounts of data and is resistant to failure of vital parts of the infrastructure. The input files are broken into parts called "blocks" by HDFS and are stored redundant across the pool of servers, thus multiple copies of data can be found in the network. In a situation where one of the nodes fails, HDFS notifies it and creates new copies of data from these redundant data. Also, multiple copies of data avoid bottlenecks in the system.

2. Related Work

A Hadoop cluster is composed of two parts: Hadoop Distributed File System and MapReduce. A Hadoop cluster uses Hadoop Distributed File System (HDFS) to manage its data. HDFS provides storage for the MapReduce job's input and output data. It is designed as a highly fault-tolerant, high throughput, and high capacity distributed file system. In [2], the authors highlight on Parallel Genetic Algorithm for the automatic generation of test suites. The solution is based on Hadoop MapReduce since it is well supported to work also in the cloud and on graphic cards, thus being an ideal candidate for high scalable parallelization of Genetic Algorithms. The amount of images being uploaded to the internet is rapidly increasing, with Facebook users uploading over 2.5 billion new photos

each month, however, applications that make use of this data are severely lacking. In [1], it is proposed that an open-source Hadoop Image Processing Interface (HIPI) that aims to create an interface for computer vision with MapReduce technology. HIPI abstracts the highly technical details of Hadoop's system and is flexible enough to implement many techniques in current computer vision literature.

3. MapReduce Techniques

3.1 Data placement in Heterogeneous cluster

In heterogeneous clusters, the capacity of computation of the nodes can be significantly different. A node with a high computing capacity can process data in a short span of time compared to the slow processing node. This algorithm has two parts: first, to distribute files among heterogeneous clusters, second, is used to reorganize the fragments and to solve the data skew problems.

3.1.1 Initial data placement

[4] Starts with dividing the large input size in fragments of even size. Then the placement algorithm distributes these fragments among the nodes according to their capacities. It distributes fragments in such a way that all the nodes complete the given task at the same time. Also the disk storage of high capacity nodes is more. Hence, the high capacity nodes are expected to do more work. It has been observed that the computing capacity and along with the response time is stable for certain Hadoop applications. This is because the response time is directly proportional to the input data.

3.1.2 Data redistribution

But the distribution using the above algorithm might be disrupted due to the following reasons: new data blocks might be added or deleted from the existing inputs, or new nodes can be added in the existing clusters. Therefore, in this algorithm it distributes the data according to computing ratio of the nodes. Here, in the algorithm, the data distribution server collects the information regarding network topology and disk storage capacity. Next, it creates two lists: first where the total number of local fragments exceeds its computing capacity and the second, where the number of local fragments is less than the capacity of the node. The former list is called over utilized node list, while the second is called as underutilized node list. Now, the data redistribution server moves the fragments from the over utilized list to the underutilized list. The nodes in over-utilized lists are source node whereas the nodes in the underutilized lists are destination nodes during migration of fragment files. This process is repeated until the total number of local fragments in each node matches its speed as measured by computing ratio.

3.2 A parallel genetic algorithm

The parallel genetic algorithm [2] is used for testing. It takes as input the software to be tested and a set of test cases. The individual fitness evaluation is carried out in parallel keeping in mind the MapReduce approach. As an output a test suite covering as many branches of the software as possible is produced. There are three models of parallelization: Global parallelization model, Island model, Grid model.

3.2.1 Global parallelization model

This model is also called as fitness evaluation model. There exists a master node which manages the population. The master node distributes the individuals among the slave nodes which compute only the values of the individuals. Thus here, the master node is the controlling node. The individual fitness evaluation is independent from the rest of the population.

3.2.2 Island model

In the island model, the population is divided into several subpopulations which are located on several nodes across the network. The genetic algorithm is carried out on each subpopulation. Different sub-populations can exchange information by sending one of their individuals to another subpopulation and vice versa. The main advantage of this method is that each subpopulation can explore different parts of the search space. In addition, the migrating 1individuals provide some diversity in the otherwise converging population.

3.2.3 Grid model

In the grid model, each individual is assigned one node. The genetic algorithm is applied simultaneously on all nodes. Parallel evaluation occurs, wherein local selection and genetic selection to a small neighboring area is done. The nodes can exchange information through migration. However, this introduces an overhead due to frequent communication between the nodes.

3.3 Image based MapReduce using HIPI

The Hadoop MapReduce techniques currently available are not suitable for processing images, although they may handle standard input and output data efficiently. This is because of the inconvenience of distributing image files across the cluster. To do this, the user has to pass the image as a string and then decode each image in each map task, in order to access pixel information. This proves to be very inefficient. But, using the technique in [1], HIPI (Hadoop Image Processing Interface), the Image Bundle data type is used as input. The images are distributed such that there is locality between the mapper machine and the machine where the image is stored. The user creates the InputFormat and RecordReader classes that describe how the MapReduce job distributes the input and what information is sent to each machine.

However, here, the solution itself provides users with both the classes, thus reducing inconvenience. Also, all the information related with the images, such as tags, descriptions, are handled, and the images are brought to the user as float images. This enables the users to directly access the pixel information. Hence tasks such as calculating the mean value of all pixels can be done effortlessly. Sometimes, the header and other information is needed without having to access the pixels. This utility is also provided by isolating such information from the pixel information.

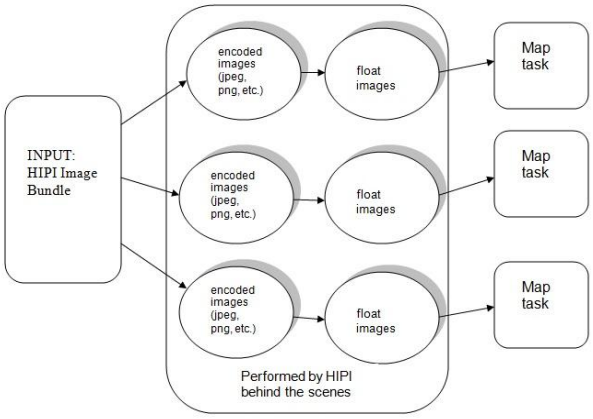

Fig.2 HIPI Frame Work

3.4 Distributed Hadoop Mapreduce On Grid

The HOG architecture proposed in [8] comprises of three components: grid submission and execution, the HDFS and the MapReduce framework.

3.4.1 Grid submission and execution

Condor and Glidein WMS are the generic frameworks used to manage the grid submission and execution. Condor is used to manage the submission and execution of the Hadoop worker nodes whereas Glidein WMS is used to allocate nodes on remote sites transparently to the user. A user can request Hadoop worker nodes to run on the grid. In this way the number of nodes can grow or shrink. There is a requirement list which restricts the Hadoop worker nodes to run on only the specified sites. The entire list of these sites is given in [8]. The executable which is specified in the condor submit file is a simple shell wrapper script that will initialize the Hadoop worker node environment. Following these steps are conducted in order to start the Hadoop worker node by the wrapper script:

1) Initializing the OSG operating environment
2) Downloading the Hadoop worker node executables
3) Extracting the worker node executables and set late binding configurations
4) Starting the Hadoop daemons
5) When the daemons shut down, cleaning up the working directory.

3.4.2 HDFS on the Grid

Creating and maintaining a distributed filesystem on a disparate set of resources is challenging. In traditional Hadoop, the datanode contacts the namenode and reports its status including information on the size of the disk on the remote node and how much is actually available for Hadoop to store. In this technique, sites are detected and separated by the reported hostnames of the worker nodes. Since the worker nodes need to be publicly addressable, they will probably have DNS names. In order to address these simultaneous preemptions, both site awareness and increased replication are used. Also, increased replication will guard against preemptions occurring faster than the namenode can replicate missing data blocks. Too many replicas would impose extra replication overhead for the namenode. The few would cause frequent data failures in the dynamic HOG environment.

3.4.3 MapReduce on the Grid

The goal of implementation in [8] is to provide a Hadoop platform comparable to that of a dedicated cluster for users to run on the grid. It should be necessary to change their MapReduce code in order to run on the proposed adaptation of Hadoop. Therefore, the authors have made no API changes to MapReduce, only underlying changes in order to better fit the grid usage model. When the grid job begins, it starts the tasktracker on the remote worker node. The tasktracker is in charge of managing the execution of Map and Reduce tasks on the worker node. When it begins, it contacts the jobtracker on the central server which marks the node available for processing. The tasktrackers report their status to the jobtracker and accept task assignments from it. In the current version of HOG, they have followed Apache Hadoop's FIFO job scheduling policy with predictive execution enabled. At any time, a task has at most two copies of execution in the system. The communication between the tasktracker and the jobtracker is based on HTTP. In the HOG system, the HTTP requests and responses are over the WAN which has high latency and long transmission time compared with the LAN of a cluster. Because of this increased communication latency, it is expected that the startup and data transfer initiations will be increased.

4. Comparative Study Of Techniques

Comparative study:

Algorithm	Computational speed	Computational time	Overhead due to communication between nodes	Use
Data Redistribution	Average	Average, time is required for both, placement of data in nodes and data redistribution.	Low, since data is distributed on heterogeneous nodes according to capacity.	Useful in applications where size of data to be distributed on the nodes is varying.
Parallel Genetic	High, mapping is done parallel on different mappers	Low	Low, there is little or no communication between mappers	Useful rapid data processing is to be done
HIPI	Average	Average, user has little to do since parallelizing and sending float images is done by HIPI	Low	Specifically used for processing of data containing images.

Table 1: comparative study

5. Conclusion

Big data and associated technologies can bring significant benefits to the business, but as the use of these technologies grows, it will become difficult for organizations to tightly control all of the many forms of data used for analysis and investigation. MapReduce can be a good approach on grid computing and image retrieval to deal with such a big data problem.

6. References

[1] Chris Sweeney Liu Liu Sean Arietta Jason Lawrence, "HIPI: A Hadoop Image Processing Interface for Image-based MapReduce Tasks", University of Virginia.

[2] Linda Di Geronimo, Filomena Ferrucci, Alfonso Murolo, "A Parallel Genetic Algorithm Based on Hadoop MapReduce for the Automatic Generation of JUnit Test Suites",2012 IEEE Fifth International Conference on Software Testing, Verification and Validation.

[3] Avrilia Floratou University of Wisconsin–Madison Jignesh M. Patel University of Wisconsin–Madison Eugene J. Shekita IBM Almaden Research Center, "Column Oriented Storage Techniques for MapReduce".

[4] 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", Department of Computer Science and Software Engineering Auburn University.

[5] Apache Hadoop Map Reduce, http://hadoop.apache.org/mapreduce/

[6] Max Grossman, Department of Computer Science Rice University,"HadoopCL: MapReduce on Distributed Heterogeneous Platforms through Seamless Integration of Hadoop and OpenCL", 2013 IEEE 27th International Symposium on Parallel & Distributed Processing Workshops and PhD Forum.

[7] Tomašić, A. Rashkovska and M. Depolli, Jožef Stefan Institute/Department of Communication Systems, "Using Hadoop MapReduce in a Multicluster Environment", MIPRO 2013, May 20-24, 2013, Opatija, Croatia.

[8] Chen He, Derek Weitzel, David Swanson, Ying Lu, Computer Science and Engineering, University of Nebraska – Lincoln "HOG: Distributed Hadoop MapReduce on the Grid", 2012 SC Companion: High Performance Computing, Networking Storage and Analysis.

[9] GuoweiWang School of Computer Science and Technology Henan Polytechnic University "A Two-phase Execution Engine of Reduce Tasks In Hadoop MapReduce", 2012 International Conference on Systems and Informatics (ICSAI 2012).