Parallel Data Mining and Assurance Service Model Using Hadoop in Cloud

Aditya Jadhav, Mahesh Kukreja

E-mail: aditya.jadhav27@gmail.com & mr_mahesh_in@yahoo.co.in

***Abstract* : In the information industry, huge amount of data is widely available and there is an imminent need for turning such data into useful information. This need is fulfilled by the process of exploration and analysis, by automatic or semi-automatic means, of large quantities of data provided by Data Mining. In case of a single system with few processors, there are restrictions on the speed of processing as well as the size of the data that can be processed at a time. The speed as well as the limit on the size of the data to be processed can be increased if data mining is carried out in parallel fashion with the help of the coordinated systems connected in LAN. But the problem with this solution is that LAN is not elastic, i.e. the number of systems in which the work is to be distributed on basis of the size of the data to be processed cannot be changed. Our main aim is to distribute data to be analyzed in various nodes in cloud. For optimum data distribution and efficient data mining as per user's desire, various algorithms must be implemented.**

I. INTRODUCTION

1.1 Cloud:

Definition of cloud computing is based on following five attributes: Multitenancy (shared resources), massive scalability, elasticity, pay as you go, and self provisioning of resources.

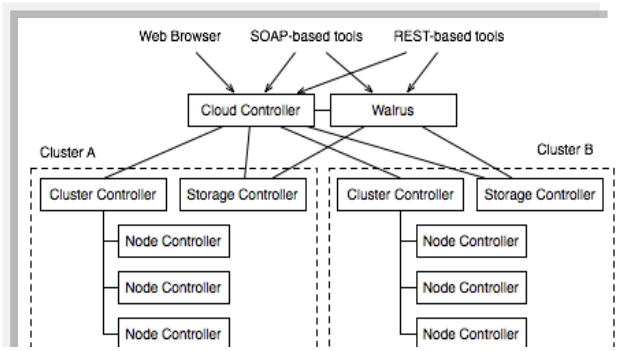
1. Multitenancy (shared resources): The ability to share resources at the network level, host level and application level is provided by Cloud computing.
2. Massive scalability: The ability to scale to tens of thousands of systems, as well as the ability to massively scale bandwidth and storage space is another advantage that Cloud computing provides.
3. Elasticity: Computing resources can be rapidly increased or decreased as needed, as well as released for other uses when they are no longer required.
4. Pay as you go: Remittance for only the resources actually used and for only the time used must be done.

1.2 Virtualization

In computing, the creation of a virtual (rather than actual) version of something, such as a hardware platform, operating system, a storage device or network resources is known as Virtualization. Virtualization can be viewed as part of an overall trend in enterprise IT that includes autonomic computing, a scenario in which the IT environment will be able to manage itself based on perceived activity, and utility computing, in which computer processing power is seen as a utility that clients can pay for only as needed. To centralize administrative tasks while improving scalability and workloads is the usual goal of virtualization.

II. EUCALYPTUS

Elastic Utility Computing Architecture for Linking Your Programs to Useful Systems - is an open-source software infrastructure for implementing "cloud computing" on clusters [1]. The current interface to Eucalyptus is compatible with Amazon's EC2 interface, but the infrastructure is designed to support multiple client-side interfaces. Commonly available Linux tools and basic Web-service technologies are used to implement Eucalyptus, making it easy to install and maintain. The creation of on-premise private clouds, with no requirements for retooling the organization's existing IT infrastructure or need to introduce specialized hardware is facilitated



The above diagram shows different components that make a Eucalyptus cloud cluster. The components are Cloud Controller (CLC), Walrus, Cluster Controller (CC), Node Controller (NC) and Storage Controller (SC).

Eucalyptus Components:

1. Cloud Controller (CLC) - Exposing and managing the underlying virtualized resources (machines or servers, network, and storage) via user-facing APIs is the responsibility of CLC. Currently, a well-defined industry standard API (Amazon EC2) is exported via a Web-based user interface.
2. Walrus - Scalable "put-get bucket storage" is implemented by Walrus. Interface compatibility with Amazon's S3 (a get/put interface for buckets and objects) is present that provides a mechanism for persistent storage and access control of virtual machine images and user data.
3. Cluster Controller (CC) - The execution of virtual machines (VMs) running on the nodes and management of virtual network between VMs and between VMs and external users is facilitated by Cluster controller.
4. Storage Controller (SC) - A block-level network storage that can be dynamically attached to VMs is provided by Storage controller. Amazon EBS semantics is supported by the current implementation.
5. Node Controller (NC) - The VM activities, including the execution, inspection, and termination of VM instances are controlled by the Node controller (through the functionality of a hypervisor.)

III. HADOOP PLATFORM FOR DISTRUBUTED COMPUTING

Applications that process vast amounts of data can be written and executed with the help of Hadoop software platform. It includes: MapReduce for offline computing engine and HDFS – Hadoop distributed file system HBase (pre-alpha) for online data access.

Apache Hadoop framework supports execution of data intensive applications on clusters built of commodity hardware [2]. Hadoop is derived from Google's MapReduce and Google File System (GFS). Similar to GFS, Hadoop has its own Hadoop File System (HDFS). Hadoop enables users to store and process large volumes of data and analyze it in ways not previously possible with less scalable solutions or standard SQL-based approaches [3].

Features of Hadoop are :

1. Scalability: Reliable storage and processing of petabytes of data is provided by Hadoop.
2. Economy: Data is distributed and processed across clusters of commonly available computers.
3. Efficiency: By distributing the data, it can be processed on the nodes where the data is located.
4. Reliability: Multiple copies of data are automatically maintained and failed computing tasks are automatically redeployed.

A single master and multiple worker nodes are included in a small Hadoop cluster. A JobTracker, TaskTracker, NameNode, and DataNode are contained in the master node. Both DataNode and TaskTracker can be executed on slave or worker node, though it is possible to have data-only worker nodes, and compute-only worker nodes; these are normally only used in non-standard applications. JRE 1.6 or higher is required. The standard start up and shutdown scripts require SSH to be set up between nodes in the cluster.

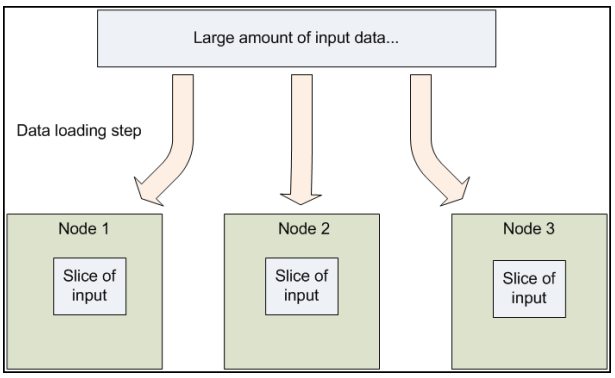
In a larger cluster, the HDFS is managed through a dedicated NameNode server that hosts the filesystem index, and a secondary NameNode that can generate snapshots of the namenode's memory structures, so preventing filesystem corruption and reducing loss of data. Similarly, job scheduling can be managed by a standalone JobTracker server. In clusters where the Hadoop MapReduce engine is deployed against an alternate filesystem, the NameNode, secondary NameNode and DataNode architecture of HDFS is replaced by the filesystem-specific equivalent.

3.1 Data Distribution

In a Hadoop cluster, data is distributed to all the nodes of the cluster as it is being loaded in. Large data files are split into chunks, by the Hadoop Distributed File System (HDFS), which are managed by different nodes in the cluster [4]. In addition to this, each chunk is replicated across several machines, so that a single

machine failure does not result in any data being unavailable. In response to system failures, data is re-replicated by an active monitoring system resulting in partial storage. Even though the file chunks are replicated and distributed across several machines, a single namespace is created that allows the contents to be universally accessible.

Data is conceptually record-oriented in the Hadoop programming framework. Individual input files are broken into lines or into other formats specific to the application logic. A subset of these records is processed by a process running on each node in the cluster. These processes are then scheduled in proximity to the location of data/records using knowledge from the distributed file system. Since files are spread across the distributed file system as chunks, a subset of the data is operated upon by a compute process running on each node. The data that should be operated on by a node is chosen based on its locality to the node: most data is read from the local disk straight into the CPU, alleviating strain on network bandwidth and preventing unnecessary network transfers. This strategy of moving computation to the data, instead of moving the data to the computation allows Hadoop to achieve high data locality which in turn results in high performance.



The above diagram shows that large data is split into pieces and loaded into different nodes in the cluster.

IV. DATA MINING

In the information industry, there is wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge in recent few years. The information and knowledge gained can be used for applications ranging from business management, production control and market analysis to engineering design and science exploration. Data mining can be viewed as a result of natural evolution of information technology for meeting this need of knowledge discovery.

Data mining is the process of exploration and analysis, by automatic or semi-automatic means, of large quantities of data, in order to discover meaningful patterns and rules. Data mining is an essential step in the process of Knowledge Discovery in databases.

4.1 k-means Method:

The k-means algorithm is used for partitioning where each cluster's centre is represented by the mean value of the objects in the cluster.

The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intra-cluster similarity is high but the inter-cluster similarity is low. Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's centroid or centre of gravity [5].

Input:

k: the number of clusters,

D: a data set containing n objects.

Output: A set of k clusters.

Method:

(1) Arbitrarily choose k objects from D as the initial cluster centers;

(2) Repeat

(3) (Re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;

(4) Update the cluster means, i.e., calculate the mean value of the objects for each cluster;

(5) Repeat until no change

V. TESTING & ANALYSIS

5.1 Eucalyptus Cloud Testing:

The following installations and configurations must be carried out to successfully deploy a Hadoop cluster in Eucalyptus cloud:

1. Installing & configuring Eucalyptus [6]
2. Configuring Eucalyptus Machine Images (EMI) [7]
3. Installing & configuring Apache Hadoop in EMI [8]
4. Installing & configuring Cloudera Hadoop in EMI [9]
5. Installing & configuring Mahout in EMI

5.2 Running Virtual machines & Executing K-means:

The Hadoop cluster consists entirely of virtual machines. A single master node and multiple data or slave nodes are present. To setup a cluster, required amount of virtual machines are booted either through the euca2ools CLI or the ElasticFox Firefox extension. Once the machines are booted and ready to run, the private key is copied to the master node. This private key provides passwordless SSH login between the master & the slave nodes. The *hosts* file on the VMs is updated with IP address/hostname pairs of master and slave nodes.

For the execution of K-means program, the first step is to start & prepare a Hadoop cluster as mentioned above. Next, the text data must be copied to the master node.

The next step is to format the Hadoop Namenode. Formatting the Namenode is required to create Hadoop Distributed File System (HDFS). After HDFS is created, Hadoop daemons are started.

Namenode, SecondaryNamenode & Jobtracker are executed on the master. And Datanode & Tasktracker are started on the slaves. Since all the nodes run in a virtual environment, it takes some time for all the daemons to boot completely & start communicating with each other. This time may range from 3-5 minutes. To verify if all the daemons are up & running, jps command can be used. After all the daemons start executing, the text data must be copied from local filesystem to HDFS. This data is copied to /user/hdfs folder on the HDFS.

The files that are copied to HDFS cannot be directly processed by K-means application. The text data must be converted into vectors [10]. There are two steps to prepare this data:

1. Converting data into SequenceFile

2. Converting SequenceFile into Vectors

After sparse vectors are created, the next step is to use the vectors as input and execute K-Means algorithm [11].

VI. RESTRICTIONS & LIMITATIONS:

1. Ubuntu instances (containing Hadoop + Mahout) cannot resolve each other through the nameserver that Eucalyptus provides. To circumvent this issue, config_all.sh script builds a hosts file containing the IP addresses/hostname pairs of the machines in the Hadoop cluster. This hosts file is then populated to all the virtual machines which are part of the Hadoop cluster.
2. Eucalyptus fails to provide a proper in-communication between VMs based on their external IP. For instance, assume a VM with 172.19.1.2 and 192.168.1.100 as its internal and external IP and another VM with 172.19.1.3/192.168.1.101 as its IPs. The first VM can only contact the other VM by contacting the internal, 172.19.1.3, IP but contacting through the external IP, 192.168.1.101, will give a timeout
3. Booting many instances at the same time increases crash risk of VMs. When Eucalyptus receives requests of booting several instances at the same time, there is a risk that the VMs either kernel panic or fail to properly inject the SSH key. If a high amount of (greater than or equal to three) instances are booting simultaneously then it is more likely that an instance will fail to boot. Requesting an instance one by one minimizes the risk, but increases the time to completely boot all the instances in a cluster.
4. Networking to a new instance is never instantly created. Even though the iptable rules and bridging is properly setup the instances are not instantly accessible. They can respond to ping, but the time it takes to access an instance through SSH usually ends at 2-4 minutes.
5. Datanode daemons don't stay up for a long time. We tried using different versions of Hadoop (0.20.2, 0.20.203.0 & 1.0.2), but all of them failed to keep the Datanode alive for more than 5 minutes. The Datanode logs showed that they failed to contact the Namenode. After the Datanodes were dead, the Namenode went down after a couple of minutes. To work around this issue, we used Cloudera's CDH3 distribution. While using Cloudera's Hadoop distribution, the connection between the Namenode & Datanodes was pretty stable. Within a couple of minutes, the Datanodes could connect to the Namenode & they would remain in a running state for a long time. But the Tasktracker (running on the slaves along with Datanodes) gave problems. It would immediately go down without updating the log file. The remaining Hadoop processes were still in the running state. But without Tasktracker, no tasks could be run.

VII. CONCLUSION

As the need of data analysis increasing day by day it's a good idea to find for more optimized and efficient way to perform this task. Distributed systems are useful for the fast and efficient data processing and main benefit of cloud is scalability

so making use of cloud and distributed we can get benefits of both under one hood.

While Hadoop is designed to be used on top of physical servers, testing has shown that running Hadoop in private cloud supplied by Eucalyptus is viable. Using virtual machines gives the user the ability to supply more machines when needed, as long as it is not reaching the physical upper limits of the underlying host machines. While setting up the cloud and Hadoop can prove problematic at first, it should not pose a problem to someone experienced in scripting, command line interfaces, networking and administration in a UNIX environment.

Using Hadoop in a virtual cluster provides an added benefit of reusing the hardware to something completely different when no MapReduce job is running. If the cloud contains several different images, it is quite viable to use a private cloud as a mean to give more computing power if needed and use the hardware to something else when it is not.

VIII. REFERENCES

[1] Eucalyptus. The Eucalyptus Open-source Cloud-computing System. http://open.eucalyptus.com/ documents / ccgrid2009.pdf

[2] Hadoop Wiki http://wiki.apache.org/hadoop/

[3] Dell. Introduction to Hadoop http://content.dell.com/ us/en/business/d/business~solutions~whitepapers~en /Documents~hadoop-introduction.pdf.aspx

[4] Storage Conference. The Hadoop Distributed File System http://storageconference.org/ 2010/ Papers/ MSST/Shvachko.pdf

[5] A Tutorial on Clustering Algorithms. K-Means Clustering http://home.dei.polimi.it/matteucc/ Clustering/ tutorial_html/kmeans.html

[6] International Journal of Computer Science Issues. Setting up of an Open Source based Private Cloud http://ijcsi.org/papers/IJCSI-8-3-1-354-359.pdf

[7] Eucalyptus. Modifying a prepackaged image http://open.eucalyptus.com/participate/wiki/modifying-prepackaged-image

[8] Michael G. Noll. Running Hadoop On Ubuntu Linux (Single-Node Cluster) http://www.michael-noll.com/tutorials/running-hadoop-on-ubuntu-linux-single-node-cluster/

[9] 8K Miles Cloud Solutions. Hadoop: CDH3 – Cluster (Fully-Distributed) Setup http://cloudblog.8kmiles.com/2011/12/08/hadoop-cdh3-cluster-fully-distributed-setup/

[10] Apache Mahout. Creating Vectors from Text https://cwiki.apache.org/MAHOUT/creating-vectors-from-text.html

[11] Amgad Madkour Blog. KMeans Clustering Using Apache Mahout http://amgadmadkour.blogspot.in /2012/07/kmeans-clustering-using-apache-mahout.html

❑❑❑