Cloud Computing and Big Data analysis using Hadoop on a Eucalyptus cloud

Abhishek Dey

Abstract:

Cloud computing is the modern and advanced variety of distributed computing where we distribute our resources or deploy our software over a network as a Service. In this work, the main focus is to build an open source EUCALYPTUS cloud ecosystem [1] to realize how we can distribute our resources from one computer or server to many nodes. This is known as Infrastructure as a Service (IaaS) [2]. This not only ensures cost reductions in up gradation of commodity hardware or low resource powered computers; but comes with a greater reliability, fault tolerance, and improved security. Also, the work deals with implementation and analyzes the importance of a software framework for the cloud ecosystem which helps us to solve or process huge data sets [3] (usually greater than 20Petabytes) using the classical MapReduce paradigm. In order to achieve this, an open source framework called HADOOP has been used which can process huge data sets over a network in few seconds (62 seconds to process 1TB) [4] using commodity hardware instead of costly and powerful dedicated servers.

Introduction:

With increase in data every day, we are coming across a problem, i.e., How can we process such huge data sets in a couple of seconds as companies like Google or Facebook have to deal with above 500PB of data. Normal mainframes or servers using DBMS will fail to process such huge amount of data. So, Google came up with the introduction of Google MapReduce and GFS (Google File System) [5], in which they distributed the workload of processing these huge data sets among various commodity hardware machines over a network. This not only resulted in unbelievable reduction in processing time of Big Data but also the system was completely fault tolerant. Yahoo and Apache developed HADOOP as an open source version of the same MapReduce algorithms and HDFS (Hadoop File System) [6] as an alternative to GFS. Hadoop replicates the same data 3 times and distributes the pieces (usually of 64MB or 128MB) [7] to several systems connected in the network. So, if one system goes down other 2 replicas are available to suffice the need. So, it's cheap, robust and fault tolerant. This can easily be achieved using a EUCALYPTUS cloud for optimal sharing or distribution of resources from the resource pool among the various machines.

EUCALYPTUS Cloud:

Eucalyptus is an open source software used to deploy private or public clouds in an IT environment from raw commodity hardware. The up gradation cost of an entire lab or an office is paramount and even upgrading software versions across servers requires expert skill as well as service is hampered or denied during such operations. The solution is CLOUD. Eucalyptus stands for "Elastic Utility Computing Architecture Linking Your Programs To Useful System". Eucalyptus is the world's most widely deployed software platform for on-premise Infrastructure as a Service (IaaS) clouds. It uses existing infrastructure to create a scalable, secure web services layer that abstracts compute, network and storage to offer IaaS that can be dynamically scaled up or down depending on workloads.

Components of Eucalyptus:

- **Node Controller** controls execution, inspection, and terminating of VM instances on the host where it runs.
- **Cluster Controller** gathers information about and schedules VM execution on specific node controllers, as well as manages virtual instance network.
- **Storage Controller (Walrus)** storage service that implements Amazon's S3 interface, providing a

mechanism for storing and accessing virtual machine images and user data.

- **Cloud Controller** is the entry-point into the cloud for users and administrators. It queries node managers for information about resources, scheduling decisions, and implements them by making requests to cluster controllers.

The front end comprises of CLC and CC. Back end comprises of NC that runs Xen or KVM hypervisor to support running instances as virtual machines on the resource pool of NCs. [8]

HADOOP:

Hadoop helps us to process huge data sets by distributing the replicated forms of same data into several datanodes whose information is stored in a namenode [9] server. There is a job tracker that splits the job into several tasks each of which is handled by a task tracker. The split files are fed into mappers where the mapping function works and keys and values are generated as (k,v) sets. These are shuffled and put to reducers who cumulate or combine the count or value of similar data sets thereby reducing redundancy of data. Also several parallel processing can be obtained by such a framework. The bottom line is that we divide the job, load it in HDFS, employ MapReduce on them, solve them in parallel, and write the cumulative results back to the HDFS. It ensures a powerful, robust and fault tolerant system that can be used to deploy huge data set processing as image processing, weather forecasting and genome grafting.

Conclusions:

Experiments can be carried out to examine whether Hadoop really makes a difference. A typical one is a Word Count example, where we provide 10 novels as input in text format and count for the frequencies of each and every word in lexicographic order and the entire job is done in a couple of seconds. This is surely the future of Data processing as we drift towards the era of Cloud computing.

References:

1. http://www.eucalyptus.com/what-is-cloud-computing
2. http://en.wikipedia.org/wiki/Infrastructure_as_a_service#Infrastructure_as_a_service_.28IaaS.29
3. http://en.wikipedia.org/wiki/Big_data
4. http://developer.yahoo.com/blogs/hadoop/posts/2009/05/hadoop_sorts_a_petabyte_in_162/
5. http://int3.de/res/GfsMapReduce/GfsAndMapReduce.pdf
6. http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/Federation.html
7. http://arunxjacob.blogspot.in/2011/04/hdfs-file-size-vs-allocation-other.html
8. http://www.change-project.eu/fileadmin/publications/Presentations/CHANGE_-_The_role_of_virtualisation_in_future_network_infrastructures_-_Warsaw_cluster_workshop_contribution.pdf
9. http://wiki.apache.org/hadoop/NameNode