Volume : 2 | Issue : 2 | Feb 2013 • ISSN No 2277 - 8160

Research Paper | Engineering

A Study on "Role of Hadoop in Information Technology era"

Vidyasagar S. D | III Sem MCA, JNN Engineering College, Shimoga-577201, Karnataka State

ABSTRACT

Nowadays, Companies need to process Multi Petabyte Datasets efficiently. The Data may not have strict schema for the large system. It has become Expensive to build reliability in each Application for processing petabytes of datasets. If there is a problem of Nodes fail every day, some of the causes of failure may be. Failure is expected, rather than exceptional. The number of nodes in a cluster is not constant. So there is a Need for common infrastructure to have Efficient, reliable, Open Source Apache License. The Hadoop platform was designed to solve problems where you have a lot of data perhaps a mixture of complex and structured data and it doesn't fit nicely into tables. It's for situations where you want to run analytics that are deep and computationally extensive, like clustering and targeting. That's exactly what Google was doing when it was indexing the web and examining user behavior to improve performance algorithms. This article has made an attempt to study its need, uses and application, thereby brought to the notice of the readers.

KEYWORDS: MapReduce, Namenode, HDFS, PB=Petabyte

Introduction

Hadoop is a "flexible and available architecture for large scale computation and data processing on a network of commodity hardware". As Hadoop is an open source framework for processing, storing and analyzing massive amounts of distributed unstructured data. Originally created by Doug Cutting at Yahoo!, Hadoop was inspired by MapReduce, a user-defined function developed by Google in early 2000s for indexing the Web. It was designed to handle petabytes and Exabyte's of data distributed over multiple nodes in parallel. Hadoop clusters run on inexpensive commodity hardware so projects can scale-out without breaking the bank. Hadoop is now a project of the Apache Software Foundation, where hundreds of contributors continuously improve the core technology. Fundamental concept: Rather than banging away at one, huge block of data with a single machine, Hadoop breaks up Big Data into multiple parts so each part can be processed and analyzed at the same time. Why Hadoop used for searching, log processing, recommendation systems, analytics, video and image analysis, data retention? It is used by the top level apache foundation project, large active user base, mailing lists, users groups, very active development, and strong development teams.

Assumptions and Goals

Hardware failure is the norm rather than the exception. An HDFS instance may consist of hundreds or thousands of server machines, each storing part of the file system's data. The fact that there are a huge number of components and that each component has a non-trivial probability of failure means that some component of HDFS is always non-functional. Therefore, detection of faults and quick, automatic recovery from them is a core architectural goal of HDFS. Applications that run on HDFS need streaming access to their data sets. They are not general purpose applications that typically run on general purpose file systems. HDFS is designed more for batch processing rather than interactive use by users. The emphasis is on high throughput of data access rather than low latency of data access. POSIX imposes many hard requirements that are not needed for applications that are targeted for HDFS. POSIX semantics in a few key areas has been traded to increase data throughput rates.

Applications that run on HDFS have large data sets. A typical file in HDFS is gigabytes to terabytes in size. Thus, HDFS is tuned to support large files. It should provide high aggregate data bandwidth and scale to hundreds of nodes in a single cluster. It should support tens of millions of files in a single instance. HDFS applications need a write-once-read-many access model for files. A file once created, written, and closed need not be changed. This assumption simplifies data coherency issues and enables high throughput data access. A MapReduce application or a web crawler application fits perfectly with this model. There is a plan to support appending-writes to files in the future. A computation requested by an application is much more efficient if it is executed near the data it operates on. This is especially true when the size of the data set is huge. This minimizes network congestion and increases the overall throughput of the system. The assumption is that it is often better to migrate the computation closer to where the data is located rather than moving the data to where the application is running. HDFS provides interfaces for applications to move themselves closer to where the data is located. HDFS has been designed to be easily portable from one platform to another. This facilitates widespread adoption of HDFS as a platform of choice for a large set of applications.

HDFS has Master/slave architecture. An HDFS cluster consists of a single Namenode, a master server that manages the file system namespace and regulates access to files by clients. In addition, there are a number of Datanodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of Datanodes. The Namenode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to Datanodes. The Datanodes are responsible for serving read and write requests from the file system's clients. The Datanodes also perform block creation, deletion, and replication upon instruction from the Namenode. Some of the goals of HDFS are Very Large Distributed File System, Assumes Commodity Hardware and Optimized for Batch Processing, Runs on heterogeneous OS.

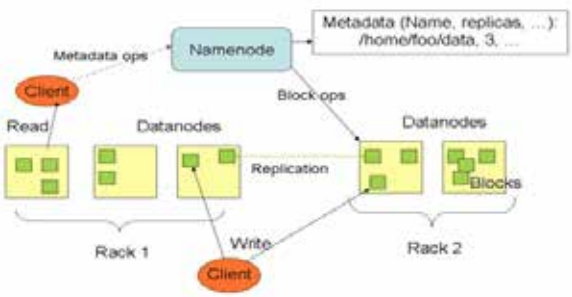


Fig: HDFS architecture

Hadoop's components

Hadoop Distributed File System: HDFS, the storage layer of Hadoop, is a distributed, scalable, Java-based file system adept at storing large volumes of unstructured data. MapReduce is a software framework that serves as the compute layer of Hadoop. MapReduce jobs are divided into two (obviously named) parts. The "Map" function divides a query into multiple parts and processes data at the node level. The "Reduce" function aggregates the results of the "Map" function to determine the "answer" to the query. Hive is a Hadoop-based data warehousing-like framework originally developed by Facebook. It allows users to write queries in a SQL-like language caled HiveQL, which are then converted to MapReduce. This allows SQL programmers with no MapReduce experience to use the warehouse and makes it easier to integrate with business intelligence and visualization tools such as Microstrategy, Tableau, Revolutions Analytics, etc.Pig Latin is a Hadoop-based language developed by Yahoo. It is relatively easy to learn and is adept at very deep, very long data pipelines (a limitation of SQL.)

HBase is a non-relational database that allows for low-latency, quick lookups in Hadoop. It adds transactional capabilities to Hadoop, allowing users to conduct updates, inserts and deletes. EBay and Facebook use HBase heavily. Flume is a framework for populating Hadoop with data. Oozie is a workflow processing system that lets users define a series of jobs written in multiple languages – such as Map Reduce, Pig and Hive -- then intelligently link them to one another. Oozie allows users to specify, for example, that a particular query is only to be initiated after specified previous jobs on which it relies for data are completed.

Flume is a framework for populating Hadoop with data. Ambari is a web-based set of tools for deploying, administering and monitoring Apache Hadoop clusters. Its development is being led by engineers from Hortonworoks, which include Ambari in its Hortonworks Data Platform.Avro is a data serialization system that allows for encoding the schema of Hadoop files. It is adept at parsing data and performing removed procedure calls. Mahout is a data mining library. It takes the most popular data mining algorithms for performing clustering, regression testing and statistical modeling and implements them using the Map Reduce model. Sqoop is a connectivity tool for moving data from non-Hadoop data stores – such as relational databases and data warehouses – into Hadoop. HCatalog is a centralized metadata management and sharing service for Apache Hadoop. BigTop is an effort to create a more formal process or framework for packaging and interoperability testing of Hadoop's sub-projects and related components with the goal improving the Hadoop platform as a whole.

Working process of Hadoop Architecture

Hadoop is designed to run on a large number of machines that don't share any memory or disks. That means you can buy a whole bunch of commodity servers, slap them in a rack, and run the Hadoop software on each one. When you want to load all of your organization's data into Hadoop, what the software does is bust that data into pieces that it then spreads across your different servers. There's no one place where you go to talk to all of your data; Hadoop keeps track of where the data resides. And because there are multiple copy stores, data stored on a server that goes offline or dies can be automatically replicated from a known good copy.

In a centralized database system, you've got one big disk connected to four or eight or 16 big processors. But that is as much horsepower as you can bring to bear. In a Hadoop cluster, every one of those servers has two or four or eight CPUs. You can run your indexing job by sending your code to each of the dozens of servers in your cluster, and each server operates on its own little piece of the data. Results are then delivered back to you in a unified whole. That's MapReduce you map the operation out to all of those servers and then you reduce the results back into a single result set. Architecturally, the reason you're able to deal with lots of data is because Hadoop spreads it out. And the reason you're able to ask complicated computational questions is because you've got all of these processors, working in parallel, harnessed together.

Hadoop implements a computational paradigm named Map/Reduce, where the application is divided into many small fragments of work, each of which may be executed or re-executed on any node in the cluster. In addition, it provides a distributed file system (HDFS) that stores data on the compute nodes, providing very high aggregate bandwidth across the cluster. Both Map/Reduce and the distributed file system are designed so that node failures are automatically handled by the framework. Hadoop Common is a set of utilities that support the other Hadoop subprojects. Hadoop Common includes File System, RPC, and serialization libraries.

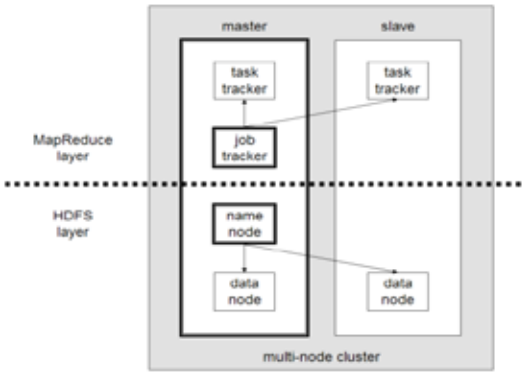


Fig: A multi-node Hadoop cluster

Requirement of Hadoop

Batch data processing, not real-time / user facing (e.g. Document Analysis and Indexing, Web Graphs and Crawling).Highly parallel data intensive distributed applications .Very large production deployments (GRID)

Process lots of unstructured data .When your processing can easily be made parallel. Running batch jobs is acceptable .When you have access to lots of cheap hardware.

Hadoop Users: The following company are the users of hadoop Adobe, Alibaba, Amazon, AOL, Facebook, Google, IBM.

Major Contributors: The following companies are the major contributors of Hadoop. They are Apache, Cloudera and Yahoo.

Conclusion

The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. Hadoop is designed to run on cheap commodity hardware, It automatically handles data replication and node failure, It does the hard work – you can focus on processing data, Cost Saving and efficient and reliable data processing. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. HDFS relaxes a few POSIX requirements to enable streaming access to file system data. HDFS was originally built as infrastructure for the Apache Nutch web search engine project. HDFS is part of the Apache Hadoop Core project.

REFERENCES

1. Apache Hadoop!(hadoop.apache.org) | 2. Hadoop on Wikipedia (http://en.wikipedia.org/wiki/Hadoop) | 3. Free Search by Doug Cutting (http://cutting.wordpress.com) | 4. Hadoop and Distributed Computing at Yahoo! (http://developer.yahoo.com/hadoop) | 5. Cloudera - Apache Hadoop for the Enterprise (http://www.cloudera.com) | 6. www.pentaho.com |