Introducing Hadoop

Distributed and Parallel Computing in Big Data

Distributed Computing

- In Distributed Computing, multiple computing resources are connected in a network and computing tasks are distributed across these resources.
- This Sharing of tasks increases the speed as well as the efficiency of the system , therefore considered faster and more suitable to process huge amounts of data in a limited time.

- Parallel Computing

Another way to improve the processing capability of a computer system is to add additional computational resources to it.This will help in dividing complex computations into subtask, which can be handled individually by processing units that are running in parallel. We call such systems as parallel systems in which multiple parallel computing resources are involved to carry out calculations simultaneously. The concept behind involving multiple parallel processors is that the processing capability will increase

Difference between Parallel Systems and Distributed Systems

1.

Distributed System- An independent , autonomous system connected in a network for accomplishing specific tasks

Parallel System – A Computer system with several processing units attached to it

2.

Distributed System- Coordination is possible between connected computers that have their own memory and CPU

Parallel System – A Common shared memory can be directly accessed by every processing unit in a network

Contd…

3.

Distributed System- Loose coupling of computers connected in a network, providing access to data and remotely located resources

Parallel System – Tight coupling of processing resources that are used for solving a single, complex problem

Techniques of Parallel Computing

- **Cluster or Grid Computing (Primarily used in Hadoop)**

Description- Cluster or Grid computing is based on a connection of multiple servers in a network. This network is known as a cluster in which the servers share the workload among them.A Cluster can be either homogeneous (comprising the same type of commodity hardware) or Heterogeneous (Consisting of different types of hardware)

Uses – A Cluster can be created even by using hardware components that were acquired a long time back to provide cost-effective storage options. The overall cost may be very high in cluster computing.

- **Massively Parallel Processing (MPP) (Used in Data Warehouse)**

Description- A Single machine working as a grid is used in the MPP platform, which is capable of handling the activities of storage, Memory, and Computing .

uses – MPP Platforms, such as EMC Greenplum and ParAccel, are most suited for high value usecases.

Contd…

- **High Performance Computing (HPC)**

Description – HPC Environments are known to offer high performance and scalability by using IMC.This technology is specially suitable for processing floating point data at high speeds.

Uses – HPC Environments can be used to develop specialty and custom applications for research and Business organizations where the result is more valuable, than the cost or where strategic importance of the project is of high priority.

Scalable Distributed Data Intensive program

- Surrounded by data –people upload videos,pictures ,cell phone , text friends,Facebook status , Twittering etc….
- Exponential growth of data first represented challenges to cutting edge businesses such as google,Yahoo , Amazon and Microsoft..
- Existing tools were inadequate to process such large data sets

- Google was the first to publicize MapReduce- a system that had used to scale their data processing needs.Other businesses were also facing similar scaling challenges
- Doug Cutting saw an opportunity and led the charge to develop an open source version of this MapReduce system called Hadoop.
- Haddop is a core part of computing infrastructure for many web companies such as Yahoo, Facebook ,LinkedIn and Twitter.

What is Hadoop?

- Hadoop is an open source framework for writing and running distributed applications that process large amounts of data.Key distinctions of Hadoop are :
- Accessible-Hadoop runs on large clusters of commodity machines or on cloud computing services such as Amazon's Elastic compute Cloud (EC2)
- Robust – can gracefully handle frequent hardware malfunctions/such failures

- Scalable – Hadoop scales linearly to handle larger data by adding more nodes to the cluster
- Simple – allows users to quickly write efficient parallel codes;even college students can quickly and cheaply create their own haddop cluster

A Hadoop cluster has many parallel machines that store and process large data sets.Client computers send jobs into this computer cloud and obtain results

Hadoop Cluster

- Hadoop cluster is a set of commodity machines networked together in one location.Data storage and processing all occur within this cloud of machines.
- Different users can submit computing jobs to Haddop from individual clients, which can be their own desktop machines in remote locations from the Hadoop cluster.

Understanding Distributed Systems and Hadoop

- Building bigger and bigger servers is non longer necessarily the best solution to large scale problems. An alternative that has gained popularity is to tie together many low end /commodity machines together as a single functional distributed systems
- Scale out (Distributed Systems) V/s Scale up (monolithic servers)
- A high end machine with four I/O Channels each having a throughput of 100 MB/Sec will require 3 hours to read a 4 TB Data set!With Hadoop same data set will be divided into smaller (typically 64 MB Blocks) that are spread among many machines in the cluster via HDFS(Hadoop Distributed File System)

Contd…

- With a modest degree of replication , the cluster machines can read the data set in parallel and provide a much higher throughput.And such a cluster of commodity machines turns out to be cheaper than one high end server!
- Hadoop focuses on moving code to data instead of vice versa-Data intensive processing
- Both data and computation exist in Hadoop cluster.

Comparing SQL Databases and Hadoop

- SQL-Structured data

  Hadoop – more of unstructured data (Text etc)
- Scale out instead of Scale up

  Scaling commercial relational databases is expensive-to run a bigger database you need to buy a bigger machine, at some point there would not be bigger enough machine available for larger data sets
- High end machines are not cost effective for many applications:a machine with four times the power of a standard PC costs a lot more than putting four PC’s in a cluster.

Contd..

- Hadoop is designed to be a scale out architecture operating on a cluster of commodity PC Machines;adding more resources means adding more machines to the cluster.

KEY/VALUE PAIRS V/S Relational

- RELATIONAL SCHEMA –not suited for various unstructured data types like Text documents , images and XML Files.Large data sets are often unstructured or semi structured.Hadoop uses key/value pairs as its basic data unit , which is flexible enough to work with the less structured data types.
- In Hadoop , data can originate in any form , but it eventually transforms into (key/value) pairs for the processing functions to work upon.

Functional Programming(MapReduce) V/s Declarative Queries(SQL)

- SQL is fundamentally a high level declarative language. You query data by stating the result you want and let the database engine figure out how to derive it. Under MapReduce you specify actual steps in processing the data, which is more analogous to an execution plan for a SQL Engine.
- Under SQL , you have query statements; under MapReduce you have scripts and codes.
- MapReduce allows you to process data in a more general fashion than SQL Queries.

Offline Batch processing v/s online transactions

- Hadoop is designed for offline processing and analysis of large scale data.It does not work for random reading and writing of few records, which is the type of load for online transaction processing.
- Hadoop is best used as a write once –read many times type of data store.

Understanding MapReduce

- Similar to pipelines and message queues in Unix, MapReduce is also a data processing model.Its greatest advantage is the easy scaling of data processing over multiple computing nodes.
- under the MapReduce model,the data processing primitives are called mappers and reducers.

Scaling a simple Program Manually

- Count the number of times each word occurs in a set of documents. In this example , we have a set of documents having only one document with only one sentence:

Do as I say, not as I do.

as 2

Do 2

I 2

Not 1

Say 1

```
define wordCount as Multiset;
for each document in documentSet {
  T = tokenize(document);
  for each token in T {
    wordCount [token]++;
                    }
                                    }
Display(wordCount);
```

For each document, the words are extracted one by one using a tokenization process.A multiset is a set where each element also has a count.This program works fine until the set of documents you want to process becomes large.e.g., you want to build a spam filter to know the words frequently used in the millions of spam emails you have received.Looping through all the documents using a single computer will be extremely time consuming.

Issues with this program

- Another flaw with this program is that wordCount are stored in memory.When processing large document sets, the number of unique words can exceed the RAM storage of machine.Looking into growing complexities of wordCount program , we need to add functionalities:

- Store files over many processing machines
- write a disk-based hash table permitting processing without being limited by RAM capacity
- Partition the intermediate data
- Shuffle the partitions to the appropriate

Scaling a same program in MapReduce

- MapReduce programs are executed in two phases: mapping and Reducing.Each phase is defined by a data processing function and these functions are called mapper and reducer.
- Mapper-MapReduce takes input dataand feeds each data element to the mapper.
- Reducer-processes all the outputs from the mapper and arrives at a final result.
- Mapper is meant to filter and transform the input into something that the reducer can aggregate over.
- Partitioning and shuffling are common design patterns that go along with mapping and reducing.

Lists and (key/value) pairs

	Input	Output
Map	<k1,v1>	list(<k2,v2>)
reduce	<k2,list(v2)>	List(<k3,v3>)

- In MapReduce framework you write applicationsby specifying the mapper and reducer.

1. The input to your applications must be structured as a list of (key/value) pairs, list(<k1,v1>).The input format for processing multiple files is usually list(<String filename,String file_content>).The input format for processing one large file, such as a log file is , list(<integer line_number,String log_event>).
2. The list of (key/value) pairs is broken up and each individual (key/value) pair <k1,v1>, is processed by calling the map function of the mapper.
3. The output of all mappers are aggregated into one giant list of <k2,v2> pairs.All pairs sharing the same k2 are grouped together into a new (key/value) pair , <k2,list(v2)>.The framework asks reducer to process each one of these aggregated pairs individually.

Word count-map reduce code

```
Map(String filename,String document){
       List<String> T = tokenize(document);
       for each token in T {
        emit ((String) token,(Integer) 1);
                                    }
                                                          }
Reduce (String token,List<Integer> values)  {
    Integer sum = 0;
     for each value in values {
        sum = sum +value;
}
  emit ((String)token,  (Integer) sum);
}
```

- Output of both map and reduce function are lists.We use a special function in the framework called emit() to generate elements in the list at a time.

Building Blocks of Hadoop

- On a fully configured cluster, running Hadoop means running a set of daemons, or resident programs, on the different servers in your network.The daemons include
  1. NameNode
  2. DataNode
  3. Secondary NameNode
  4. JobTracker
  5. TaskTracker

NameNode

- Hadoop employs a master/slave architecture for both distributed storage and distributed computation.
- The NameNode is the master of HDFS that directs the slave DataNode daemons to perform the low-level I/O tasks.Th NameNode is the book keeper of HDFS;it keeps track of how your files are broken down into file blocks, which nodes store those blocks, and the overall health of distributed filesystem.
- Function of NameNode is memory and I/O intensive

DataNode

- Each slave machine in cluster will host a DataNode daemon to perform the grunt work of the distributed file system-reading and writing HDFS blocks to actual files on the local filesystem. When you want to read or write a HDFS file, the file is broken into blocks and the NameNode will tell your client which DataNode each block resides in.Your client communicates directly with DataNode daemons to process local files corresponding to blocks.

Ex. : two data files - /user/chuck/data1,/user/james/data2

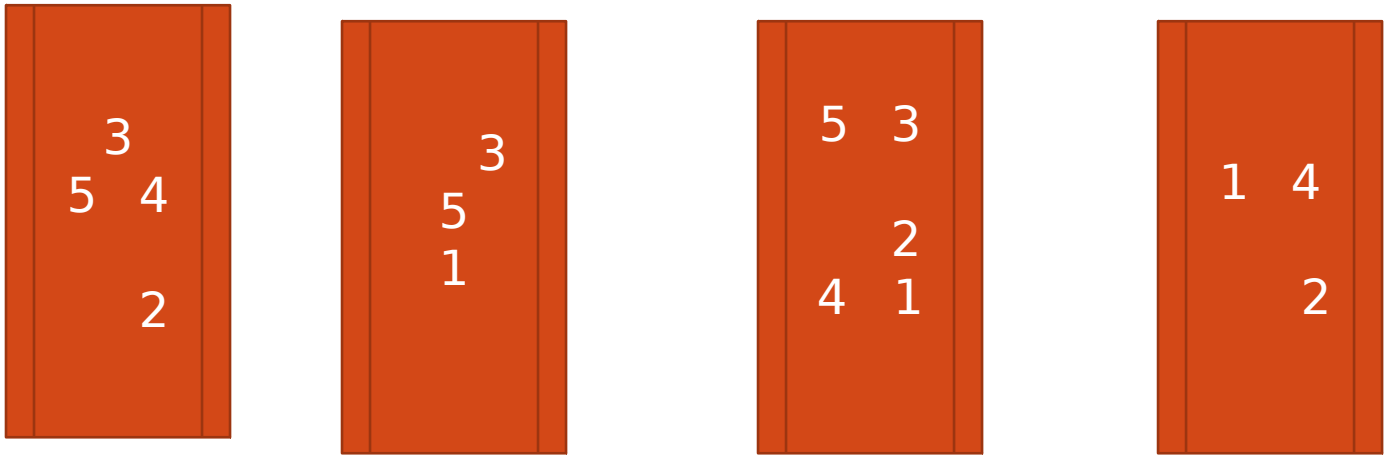
data1- takes 3 blocks, and data2 takes 2 blocks.The contents of files are distributed among the DataNodes.DataNodes constantly report to NameNodes.

NameNode keeps track of the file metadata-which files are in system and how each file is broken down into blocks.The DataNodes provide backup store of the blocks and constantly report to the NameNode to keep metadata current.

- NameNode

File Metadata
/user/chuck/data1->1,2,3
/user/james/data2->4,5

- DataNodes



Secondary NameNode(SNN)

- SNN is an assistant Daemon for monitoring the state of the cluster HDFS.Like the NameNode, each cluster has one SNN, and it typically resides on its own machine as well.SNN differs from NameNode in that this process does not receive or record any real time changes to HDFS.
- NameNode is a single point of failure for a Hadoop cluster, and the SNN snapshots help minimize the downtime and loss of data.

JobTracker

- It is the liason between your application and Hadoop.Once you submit your code to cluster, the JobTracker determines the execution plan by determining which files to process, assigns nodes to different tasks and monitors all tasks as they are running.Should a task fail, the JobTracker will automatically relaunch the task, possibly on a different node, up to a predefined limit of retries.
- There is only one JobTracker daemon per Hadoop cluster.

TaskTracker

- JobTracker is the master overusing the overall execution of a MapReduce job and Tasktrackers manage the execution of individual tasks on each slave node .
- Each TaskTracker is responsible for executing the individual tasks that the JobTracker assigns.Although there is a single TaskTracker per slave node, each TaskTracker can spawn multiple JVMs to handle many map or reduce tasks in parallel.
- Responsibility of TaskTracker is to constantly communicate with the JobTracker.If the JobTracker fails to receive a heartbeat from a TaskTracker within a specified amount of time , it will assume the TaskTracker has crashed and will resubmit the corresponding tasks to other nodes in the cluster.

Summary

- Hadoop is a versatile tool that allows new users to access the power of distributed computing.
- By using distributed storage and transferring code instead of data , Hadoop avoids the costly transmission step when working with large data sets.
- Redundancy of data allows Hadoop to recover should a single node fail
- Need not to worry about partitioning the data , determining which nodes will perform which tasks, or handling communication between nodes.