Big Data Introduction

Data is largely classified as Structured, Semi-Structured and Un-Structured.

If we know the fields as well as their datatype, then we call it structured. The data in relational databases such as MySQL, Oracle or Microsoft SQL is an example of structured data.

The data in which we know the fields or columns but we do not know the datatypes, we call it semi-structured data. For example, data in CSV which is comma separated values is known as semi-structured data.

If our data doesn't contain columns or fields, we call it unstructured data. The data in the form of plain text files or logs generated on a server are examples of unstructured data.

The process of translating unstructured data into structured is known as ETL - Extract, Transform and Load.

What is Distributed System?

When networked computers are utilized to achieve a common goal, it is known as a distributed system. The work gets distributed amongst many computers. The branch of computing that studies distributed systems is known as distributed computing. The purpose of distributed computing is to get the work done faster by utilizing many computers.

Most but not all the tasks can be performed using distributed computing.

What is Big Data?

In very simple words, Big Data is data of very big size which can not be processed with usual tools. And to process such data we need to be distributed architecture. This data could be structured or unstructured.

Generally, we classify the problems related to the handling of data into three buckets:
Volume: When the problem we are solving is related to how we would store such huge data, we call it Volume. Examples of Volume are Facebook handling more than 500 TB data per day. Facebook is having 300 PB of data storage.

Velocity: When we are trying to handle many requests per second, we call this characteristic Velocity. The problems as the number of requests received by Facebook or Google per second is an example of Big Data due to Velocity.

Variety: If the problem at hand is complex or data that we are processing is complex, we call such problems as related to variety.

Imagine you have to find the fastest route on a map since the problem involves enumerating through many possibilities, it is a complex problem even though the map's size would not be too huge.

Big Data Introduction

Data could be termed as Big Data if either Volume, Velocity or Variety becomes impossible to handle using traditional tools.

1. CPU - Which executes instructions. CPU is characterized by its speed. More the number of instructions it can execute per second, faster it is considered.

Then comes RAM. Random access memory. While processing, we load data into RAM. If we can load more data into ram, CPU can perform better. So, RAM has two main attributes which matter: Size and its speed of reading and writing.

To permanently store data, we need hard disk drive or solid state drive. The SSD is faster but smaller and costlier. The faster and bigger the disk, faster we can process data.

Another component that we frequently forget while thinking about the speed of computation is a network. Why? Often our data is stored on different machines and we need to read it over a network to process.

While processing Big Data at least one of these four components become the bottleneck. That's where we need to move to multiple computers or distributed computing architecture.

A/B testing

A/B Testing is a process used in testing drugs to identify if a drug is effective or not.

A similar process is used to identify if a particular feature is effective or not. As you can see in the diagram, randomly selected half of the users are shown variation A and other half is shown variation B. We can clearly see that variation A is very effective because it is giving double conversions. This method is effective only if we have a significant amount of users. Also, the ratio of the users need not be 50-50.

So, A/B testing utilizes the data in product development. At Amazon, we have many such experiments running all the time. Every feature is launched via A/B testing. It is first shown to say 1 percent of users and if it is performing good, we increase the percentages.

To manage so many variations on such a high number of users, we generally need Big Data platforms.

Big Data Solution stacks.

The first and most powerful stack is Apache Hadoop and Spark together. While Hadoop provides storage for structured and unstructured data, the Spark provides the computational capability on top of Hadoop.

The second way could be to use Cassandra or MongoDB.

Third could be to use Google Compute Engine or Microsoft Azure. In such cases, you would have to upload your data to Google or Microsoft which may not be acceptable to your organization sometimes.

Hadoop
Hadoop was created by Doug Cutting in order to build his search engine called Nutch. He was joined by Mike Cafarella. Hadoop was based on the three papers published by Google: Google File System, Google MapReduce, and Google Big Table.

It is named after the toy elephant of Doug Cutting's son.

Hadoop is under Apache license which means you can use it anywhere without having to worry about licensing.

It is quite powerful, popular and well supported.

It is a framework to handle Big Data.

Started as a single project, Hadoop is now an umbrella of projects. All of the projects under the Apache Hadoop umbrella should have followed three characteristics: 1. Distributed - They should be able to utilize multiple machines in order to solve a problem. 2. Scalable - If needed it should be very easy to add more machines. 3. Reliable - If some of the machines fail, it should still work fine. These are the three criteria for all the projects or components to be under Apache Hadoop.

Hadoop is written in Java so that it can run on kinds of devices.

The Apache Hadoop is a suite of components. Let us take a look at each of these components briefly. We will cover the details in depth during the full course.

HDFS

HDFS or Hadoop Distributed File System is the most important component because the entire eco-system depends upon it. It is based on Google File System.

It is basically a file system which runs on many computers to provide a humongous storage. If you want to store your petabytes of data in the form of files, you can use HDFS.

YARN or yet another resource negotiator keeps track of all the resources (CPU, Memory) of machines in the network and run the applications. Any application which wants to run in distributed fashion would interact with YARN.

HBase

HBase provides humongous storage in the form of a database table. So, to manage humongous records, you would like to use HBase.

HBase is a kind NoSQL Datastore.

MapReduce

MapReduce is a framework for distributed computing. It utilizes YARN to execute programs and has a very good sorting engine.

You write your programs in two parts Map and reduce. The map part transforms the raw data into key-value and reduce part groups and combines data based on the key. We will learn MapReduce in details later.

Spark

Spark is another computational framework similar to MapReduce but faster and more recent. It uses similar constructs as MapReduce to solve big data problems.

Spark has its own huge stack. We will cover in details soon.

Hive

Writing code in MapReduce is very time-consuming. So, Apache Hive makes it possible to write your logic in SQL which internally converts it into MapReduce. So, you can process humongous structured or semi-structured data with simple SQL using Hive.

Pig (Latin)

Pig Latin is a simplified SQL like language to express your ETL needs in stepwise fashion. Pig is the engine that translates Pig Latin into Map Reduce and executes it on Hadoop.

Mahout

Mahout is a library of machine learning algorithms that run in a distributed fashion. Since machine learning algorithms are complex and time-consuming, mahout breaks down work such that it gets executed on MapReduce running on many machines.

ZooKeeper

Apache Zookeeper is an independent component which is used by various distributed frameworks such as HDFS, HBase, Kafka, YARN. It is used for the coordination between various components. It provides a distributed configuration service, synchronization service, and naming registry for large distributed systems.

Flume

Flume makes it possible to continuously pump the unstructured data from many sources to a central source such as HDFS.

If you have many machines continuously generating data such as Webserver Logs, you can use flume to aggregate data at a central place such as HDFS.

SQOOP

Sqoop is used to transport data between Hadoop and SQL Databases. Sqoop utilizes MapReduce to efficiently transport data using many machines in a network.

Oozie

Since a project might involve many components, there is a need of a workflow engine to execute work in sequence.

For example, a typical project might involve importing data from SQL Server, running some Hive Queries, doing predictions with Mahout, Saving data back to an SQL Server.

This kind of workflow can be easily accomplished with Oozie.

Apache Spark is a fast and general engine for large-scale data processing.

It is around 100 times faster than MapReduce using only RAM and 10 times faster if using the disk.

It builds upon similar paradigms as MapReduce.

It is well integrated with Hadoop as it can run on top of YARN and can access HDFS.

Resource Managers

A cluster resource manager or resource manager is a software component which manages the various resources such as memory, disk, CPU of the machines connected in the cluster.

Apache Spark can run on top of many cluster resource managers such YARN, Amazon EC2 or Mesos. If you don't have any resource managers yet, you can use Apache Spark in Standalone mode.

Sources

Instead of building own file or data storages, Apache spark made it possible to read from all kinds of data sources: Hadoop Distributed File System, HBase, Hive, Tachyon, Cassandra.

Libraries

Apache Spark comes with great set of libraries. **Data frames** provide a generic way to represent the data in the tabular structure. The data frames make it possible to query data using R or SQL instead of writing tonnes of code.

Streaming Library makes it possible to process fast incoming streaming of huge data using Spark.

MLLib is a very rich machine learning library. It provides very sophisticated algorithms which run in distributed fashion.

GraphX makes it very simple to represent huge data as a graph. It proves library of algorithms to process graphs using multiple computers.

Spark and its libraries can be used with **Scala, Java, Python, R, and SQL**. The only exception is GraphX which can only be used with Scala and Java.

With these set of libraries, it is possible to do ETL, Machine Learning, Real time data processing and graph processing on Big Data.