**BIG Data Introduction**

**The speed and the quantum of data that is generated by IT systems**

* The New York Stock Exchange generates about one terabyte of new trade data per day (2008).
* Google was processing 20 petabytes of data a day (2008)
* Ancestry.com, the genealogy site, stores around 2.5 petabytes of data. (2008)
* Walmart handles more than 1 million customer transactions every hour.

**Data , Data, everywhere Data(Social-Media Data)**

* YouTube users upload 48 hours of new video every minute of the day.
* Twitter sees roughly 175 million tweets every day, and has more than 465 million accounts (2012)
* Facebook adds 500 TB data daily(2012).

The volume of social-media data is getting bigger and bigger.

**Mobile Data**

* More than 5 billion people are calling, texting, tweeting and browsing on mobile phones worldwide.
* There are 924,318,927 (92 crore) mobile phones in India, 74% of the population
* There are 6,800,000,000 (680 crore) mobile phones in world which is 97% of the population

In short, there is a huge explosion of Data(PDF1 – page19-20)

**Processor or Virtual Storage**

· 1 Bit = Binary Digit  
· 8 Bits = 1 Byte  
· 1024 Bytes = 1 Kilobyte   
· 1024 Kilobytes = 1 Megabyte   
· 1024 Megabytes = 1 Gigabyte   
· 1024 Gigabytes = 1 Terabyte   
· 1024 Terabytes = 1 Petabyte   
· 1024 Petabytes = 1 Exabyte  
· 1024 Exabytes = 1 Zettabyte   
· 1024 Zettabytes = 1 Yottabyte   
· 1024 Yottabytes = 1 Brontobyte  
· 1024 Brontobytes = 1 Geopbyte

1 Terabyte = 1024 GB (Gigabytes)

1 Petabyte = 1024 Terabytes

**Why DATA is also a Big Opportunity ?** (PDF1 – page26)

* Finding out **useful trends of data from a huge data-set is more accurate**, as compared to a smaller data-set.
* Machine Learning systems can find out **interesting patterns and insights just** by observing the huge datasets. This insight might not be available in small data.
* “Confidence Level” or **Accuracy of statistical calculations are quite high** on big data-sets. The advantage of larger data-sets is due to the additional information derived from analysis of a single large set of related data
* Server logs or Web Server logs of IP addresses of machines of customers are heavily used to generate **customer-buying or customer-browsing habits** on a given shopping-site.
* A weather-condition that happens or occurs every five years cannot be tracked if you just have weather data for 2 or 3 years. Hence, weather-data for say 15 years, can help you to track such once-in-five year weather event.

**Thus, a larger data-set is better.**

**What is Big Data?** (PDF1 - page3)

* + Big data is the term for a **collection of data sets so large and complex** that it becomes **difficult to process it using old database management tools or traditional data processing technologies** like DBMS or RDBMS.
  + BIG DATA is **data that keeps coming** for example data coming from social media which keeps coming and you **cannot handle it with traditional ETL tool/databases** and need some special tools to store/process such data.
  + A common aspect of Big Data is that it is often **data that was otherwise ignored** in your business because **you did not have the capability to store,** process and analyze it, like web-server log data because it was in terabytes.

**The Three V’s or properties of Big Data**(PDF1 – page4-13)

The characteristics of Big Data are often defined as the 3 V’s:

* **Variety**: any type of **structured or unstructured data**.
* **Volume**: terabytes and petabytes (and even exabytes) of data.
* **Velocity**: data **flows in to your organization/system at increasing rates** or very high speeds.

**Types of data in Big Data**(PDF1 – page12)

Big Data includes all types of data in contrast to the popular myth that it is only unstructured data:

* **Structured:** the **data has a schema/structure**, or a schema/structure can be easily assigned to it. Example : tables
* **Semi‐structured**: has **some structure**, but typically columns are often missing or each row has it’s own unique columns. Example : XML data
* **Quasi-Structured** : This is textual data, which is quite erratic and needs a lot of time, efforts and tools to be structured. Example : Web-Click Stream data
* **Unstructured**: data that has **no structure.** Example : JPGs, PDF files, audio and video files, etc.

**Big data in various sectors**(PDF1 – page23)

**Big Data in Science & Research**

• **Astronomy:** Sloan Digital Sky Survey (SDSS) Telescope records 200 GB per day. When the Large Synoptic Survey Telescope, successor to SDSS, comes online in 2016 it is anticipated to record 30 petabyte per day using Big data technology

• **Medical:** Decoding the human genome or DNA originally took 10 years to process, now it can be achieved in less than a week

• **Research:** Research on public data like “Tobias Preis et al”. used Google Trends data to demonstrate that Internet users from countries with a higher GDP are more likely to search for information about the future than information about the past.

**Big Data in Government/Politics**

• Big data analysis played a large role in Barack Obama's successful 2012 re-election campaign in USA’s president election and Mr. Modi’s win in 2014 in India. BIG DATA which was collected earlier, helped in finding out information like people from which area would be interested in which topic, and which would be the topics which should not be touched upon during the election campaigns.

• In India the UID, Unique Identification or “Aadhar” project started to collect the huge amount of biometric data about Indian citizens. Technical Platform of UID is built on Hadoop and other open source software.

**Big Data in Private Sector**

• **E-commerce sectors like EBay.com** uses two data warehouses of around 7.5 petabytes and 40 petabytes. And a 40PB Hadoop cluster for search, consumer recommendations, and merchandising.

• **Amazon.com** handles millions of back-end operations every day, as well as queries from more than half a million third-party sellers. As of 2005 they had the world’s three largest Linux databases, with capacities of 7.8 TB, 18.5 TB, and 24.7 TB.

• **Flipkart.com uses Hadoop Ecosystem** technologies to handle their user engagement platform.

**Big data Examples/Use-Cases**(PDF1 – page24-25)

Upcoming Big Data Examples (Use Cases)

* + Clickstream data
  + Sentiment data
  + Geolocation data
  + Server Log data
  + Machine and Sensor data
  + Data warehouse optimization

**Clickstream data**

A clickstream is the **stream of clicks when a user navigates through a company’s website**. It is **obtained from the web-server logs**, which can be **captured and stored and analyzed**.

Here are some of the ways that clickstream data can provide benefit to an organization:

• Basket Analysis

* **examining things as customer interests**,
* when customers bought Product X, what **common paths** did they take to get there

• Next Product to Buy Analysis

* this type of analysis looks at **correlation in purchases**

• Allocation of Website Resources

* We know the hottest**(most visited)** and coldest paths**(least visited)** on the site and can assign resources accordingly.

• Granular Customer Segmentation

* a company can discover the trends of a particular segment or group of customers
* cover and gain insight on how particular **segments and micro-segments of customers** are using the site, and how to best cater to them.

• A 360 degree view of the customer

Online retailers want to find out what shoppers are doing on their sites -- **what pages they visit, where they linger, how long they stay, and when they leave**. This all is unstructured clickstream data. A unified view of this data along with customer’s structured data can provides us invaluable knowledge about how to approach and sell to a particular customer or customer-category.

**Sentiment data**

• Understand how your **customers feel about your brand and products** and your competitor’s brand and their products, may be right now, may be in the past

• Based on this information a company can **custom-target their marketing and ad campaigns**.

An example case study conducted by Hortonworks using Hadoop was at the **launch of movie Iron Man 3, by using Twitter data**.

The steps that were performed were:

• Used Flume to get the Twitter feeds into HDFS.

• Used HCatalog to define a shareable schema for the data.

• Used Hive to determine sentiment.

• Used an Excel bar graph to visualize the volume of tweets.

• Used MS PowerView to view sentiment by country on a map.

They used all the above mentioned hadoop tools so that they could do the sentiment analysis.

**Geolocation data**

• Analyze **location-based data to manage operations** where they occur.

• Geolocation data gives organizations the **ability to track every moving aspect of their business**, be it the objects or the individuals.

• Retailers are **using moving Wi-Fi signals** from phones to track their customer’s movements through their stores for the purpose of improving store layouts.

• Fujitsu, created a **real-time criminal activity map** using the geolocation data from Twitter streams, creating a map that people can use to see where trouble areas are at any given time.

**Server Log data**

• All **internet traffic and network traffic data gets stored in server logs**. Servers can be LAN servers, Web application server, Security servers, etc.

• Server log data can be used for n**etwork usage, security threats, and compliance**.

• Admins are able to **load their server logs into Hadoop** using Apache Flume, building a repository that they can use for these analysis.

**Machine and Sensor data**

• Hadoop is a very attractive store for **sensor data due to the ability to dump** so much and then **use analysis tools to extract correlations** that give insights on operations.

• These sensors monitor and track very specific things, such as temperature, speed, location.

**Data warehouse optimization**

• A large company, hoping to boost the **efficiency of its enterprise data warehouse**, will look to **archive data** that might be stored more cost effectively on a Hadoop platform.

• A Hadoop cluster is implemented to **offload less frequently** used data from the existing data warehouse.

• The company **saves on storage costs and speeds-up query performance** and access to their analytic data mart.

• Some ETL tasks are also offloaded to Hadoop.

Example : Big Data at Facebook - implementing it’s Datawarehouse in hadoop

Initially, data warehousing at Facebook was performed entirely on an Oracle instance. As part of this investigation, they deployed a relatively small Hadoop instance and started publishing some of their core datasets into this instance.

**Big data Comparison with Business Intelligence and Analytics**

Analytics Tools or Analytics Front end layer

Big Data

Internal Data

Enterprise DW

D/W

External Data

Reporting Tools or BI front end layer

Big Data v/s Others

* Business Intelligence is an **umbrella-term used for end-to-end decision science process**.
* In a typical BI system, as shown above, Big Data, if used, will have it’s own place and will replace none of them. **They all along with Big Data will work together.** So, all these technologies like Data Warehouse, Internal Data, External Data, Analytics Front End Layer, BI Front End Layer Database and Big Data **will all work in synchronization and not as an alternative to one another**.
* **Internal data** is the data which comes from your **company’s IT system, CRM systems, SCM systems, Product systems,** etc..
* **External data** is social media data or sentiment analysis data or **data which you buy/retrieve from data vendors** who do some research and provide you with market data.
* **Based on the type of data and volume of data** you would need to decide as to whether this internal data and external **could be better processed by a Data warehouse System or by Big Data system**.
* Your Data warehouse system and the Big data system could also talk to each other.
* BI Reporting Layer Database, also known as the **Enterprise Data Warehouse**(tools Micro-strategy or Teradata) which **stores the summarized data** or all the data in the final format.
* **BI front end** is used for **reporting**.
* **Analytics Front end layer** is mostly used for **predictions, analyses** and other algorithmic techniques to help take **strategic-level decisions**.
* Your **BI Front End Layer(or Reporting tools) and Analytics Front End Layer(or Analytics tools) are connected to your Enterprise Data Warehouse**. But, they **could also be directly connected**, if needed, to your **Big Data** .
* Traditional **DW which holds your company’s structured data** generated over several years.
* **Big data** would help you to efficiently **store the unstructured data** got from external systems.
* **Big data** would be essentially used for **data-lakes and/or processing unstructured data.**
* **Traditional ETL tools** can be used **transfer data from DW to Enterprise DW**, and **tools like scoop** can be used efficiently to **transfer data from Big data to Enterprise DW**, as a DW and an Enterprise DW is finally an RDBMS

**Big data in Cloud**

What is Cloud?

* **Cloud** is **using the remote resources for** your **local needs**.
* You want to do **some processing or run some program**, but you **do not have sufficient resources**.
* **Virtualization** is the process of implementing this **cloud service**.

Virtual Machine

Virtual Machine

Virtual Machine

Virtual Machine

Virtualization Software

Host Operating System

Physical Hardware

* The cloud vendors create a lot of virtual machines and you connect to these virtual machines and do the processing on those virtual machines.
* The virtualization software/layer will increase the resources or processing capabilities of that virtual machine.

Cloud Delivery Models :

* Software as a Service (SaaS)
* Platform as a Service (PaaS)
* Indrastructure as a Service (IaaS)

**Software as a Service (SaaS)** :

These are **softwares which are available on the cloud** and you can go there on the cloud and use them, like billing softwares, CRM softwares like Sales Force, many Adobe products, etc.

**Platform as a Service (PaaS)**

Here, the **programming platforms** like .Net platform or java platform are **provided to you on the cloud** wherein you use the programming platform to create and deploy your own applications

**Infrastructure as a Service (IaaS)**

Here, they give you the **underlying infrastructure or hardware** and then you **install your own operating system, install you own programming platform and take care of all things starting from the operating system**.

**Big Data Cloud Providers**

**Amazon**

Amazon is the **leader in cloud service market** and they **provide many tools**. They offer big data services to customers from it’s Amazon web Services portfolio. These include the following:

Amazon Elastic MapReduce:

It is based on hadoop and you can run your MapReduce programs on the cluster provided by Amazon on which you have implemented your Big-data.

Amazon DynamoDB: This a database tool.

Amazon RedShift: This a Data warehousing tool. It is a petabyte-scale data warehousing service.

**Google big data services**

Google Compute Engine

A cloud-based capability for virtual machine computing, Google Compute Engine offers a secure, flexible computing environment.

Google Big Query

It is an OLAP system for Big Data. You can store Big Data on Google Big Query and you can query that data as you query any OLAP system or reporting system.

Google Prediction API

A set of machine learning algorithms. Analysis, predictions, and artificial intelligence are provided through APIs.

**Microsoft Azure for big data**

Azure is a Microsoft cloud service, based on Windows and SQL abstractions. Azure is a set of development tools for media services, and mobile services in a PaaS offering.

For Big Data, Microsoft has added Windows Azure HDInsight. Azure HDInsight for Big data is thus cloud service for Big data provided by Microsoft.

**Big Data High Level Architecture**

At a higher level, Big-Data Architecture is dominated by the following two :

* Brewer’s Theorem
* Map Reduce Architecture

Big data is focused more towards processing unstructured and huge amount of incoming data, whereas RDBMS is focused on processing structured data and comparatively smaller chunks of transactional data, due to the architectural differences between the two of them.

Traditional RDBMS Database’s work on ACID(Atomicity, Consistency, Isolation, Durability) properties which is a set of properties that guarantees that database transactions be processed reliably.

•Atomicity requires that each transaction be "all or nothing". For example, if an update statement was to update 500 rows, and after updating 150 rows, there was some sort of error, then the already updated 150 rows would be rolled back.

•Consistency property ensures that any transaction will bring the database from one valid state to another, regardless of whether the transaction was committed or rolled-back.

•Isolation property ensures transaction-results are not visible to other sessions until they are committed.

•Durability means that once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or any other such errors.

The above 4 are good to maintain integrity of data.

**Brewer’s Theorem**

For Bigdata(or distributed systems), we have the Brewer's theorem also known as **CAP theorem**, and it states that it is **impossible for a distributed computer system to simultaneously provide all three** of the following :

Consistency

Availability

Partition tolerance

•**Consistency** means that data is the same across the cluster, that is, **data is the same on all on the nodes of a distributed system**

•**Availability** means the ability to **access the data on a cluster or distributed system, even if a few nodes in the cluster** on which the data is spread **are down**.

•**Partition Tolerance** means that the cluster or **distributed system continues to function** even if there is a "partition" **(communications break-down) between two nodes** (both nodes are up, but can't communicate).

It says, that you need to **compromise on atleast one of the above three** to ensure the proper function of a distributed system or cluster of nodes

Thus, the Brewer’s theorem say that you can have multiple copies of your data on cheaper nodes or machines. Thus, a 100-machine cluster, wherein for every data-set you are having 10 copies. If some of the machines are down, then you still have more copies of your data available on other nodes of your cluster for read/processing-purpose

**Map Reduce Architecture**

Big data **processing architecture** is based on the Map Reduce architecture.

It is designed to be used for **data-intensive problems**. The data would be distributed on multiple machines or nodes and then, whatever **programs or processes you are going to run, need to be sent to all these nodes or machines and these programs would run all these nodes**. The output will be generated by all these machines and it would be collected by a single machine, and the end user gets the final output from this single machine. All those individual machines or nodes on which the programs ran are called MAPPERS and the final machine which gave you the output is called REDUCER.

• **"Map" step:** The master node takes the input problem, and distributes them to worker nodes. A worker node may do this again in turn, leading to a multi-level tree structure. The worker node processes the smaller problem, and passes the answer back to its master node.

• **"Reduce" step:** The master node then collects the answers to all the sub-problems and combines them in some way to form the output – the final answer .

Example :

Suppose you have 10000 rows of employees in a table called EMP. You have the following query to be solved :

Query=Select sum(sal) from emp;

2000 rows

2000 rows

2000 rows

2000 rows

2000 rows

5 rows

2000 rows each are stored on 5 nodes or machines. Then, the above query is executed on each of those 5 machines, which give 1 output row each. Hence, there would 5 output rows, which are moved to another machine. And then aggregate the 5 outputs on this machine. Thus, only 5 rows to be processed on this machine. Thus, one final out is generated on this machine, which is finally sent to the user/application which executed the above query.

Why is this architecture better than earlier architectures or What is the catch?

In Teradata we had **Massively Parallel Processing(MPP) architecture**. In the map-reduce architecture, we are r**unning the programs or queries on the same machine which is holding the data. This is known as data-locality.**But, **in the earlier MPP** system, we had to **move all the 10000 rows from the different disks to a set of parallel processors,** and then the program or query was executed on those set of parallel processors. Thus, **data movement took a lot of time in MPP systems**. Thus, map-reduce is more efficient.

**Map Reduce Example**

Calculate the number of occurrences of each words in below sentences:

Peter Piper picked a peck of pickled peppers

A peck of pickled peppers Peter Piper picked

If Peter Piper picked a peck of pickled peppers

Wheres the peck of pickled peppers Peter Piper Picked?

Thus, you can have 4 mappers here assuming the above sentences are stored on different nodes of the cluster. Each mapper will count the number of words in a given sentence. The reducers will then combine the results and then another final reducer which will combine the results of the earlier reducers. Thus,

Peter Piper picked a peck of pickled peppers Peter -1

Piper -1

picked -1

Reducer

a -1

peck -1

Map

of -1

pickled -1

peppers -1

Peter -2

Piper -2

picked -2

a -2

peck -2

of -2

Reducer

pickled -2

peppers -2

A peck of pickled peppers Peter Piper picked A -1

peck -1

of -1 peter -4

pickled -1 piper -4

Reducer

peppers -1 picked -4

Peter -1 a -3

Map

Piper -1 peck -4

picked -1 of -4

pickled -4

peppers -4

If Peter Piper picked a peck of pickled peppers If -1 if -2

Peter -1 wheres -1

Piper -1 the -1

picked -1

Map

a -1

Reducer

peck -1

of -1

pickled -1

peppers -1

If -2

Peter -2

Piper -2

Reducer

picked -2

a -2

peck -2

of -2

pickled -2

peppers -2

wheres -1

the -1

Wheres the peck of pickled peppers Peter Piper Picked? Wheres -1

the -1

peck -1

of -1

Reducer

Map

pickled -1

peppers -1

Peter -1

Piper -1

picked -1

Catch is the parallelism and **locality of data for each processor** which can help in creating a huge cluster of cheap machines which can do lot of things in less time.

**Walkthrough :**

Hadoop MapReduce is a software framework for writing applications which process vast amounts of data(multi-terabyte data-sets) in-parallel on large clusters(thousands of nodes) in a reliable, fault-tolerant manner.

A MapReduce job usually works on the **splits of the input data-set into independent chunks which become inputs to the map tasks**. These chunks are **then processed by the map tasks in a completely parallel manner**. Then, the framework **sorts the outputs of the maps**, which are **then input to the reduce tasks**. Typically both the input and the output of the job are stored in a file-system.

Input Data-Sets= File1

Hello World Bye World

File2

Hello Hadoop Goodbye Hadoop

So, the independent Chunks which become input to the map tasks =

Hello World Bye World

Hello Hadoop Goodbye Hadoop

Output of the maps =

first map emits:  
< Hello, 1>   
< World, 1>   
< Bye, 1>   
< World, 1>

The second map emits:  
< Hello, 1>   
< Hadoop, 1>   
< Goodbye, 1>   
< Hadoop, 1>

Sorting and, if needed, then combining the output of the maps =

The output of the first map, sorted

< Bye, 1>   
< Hello, 1>   
< World, 1>

< World, 1>

and then combined:  
< Bye, 1>   
< Hello, 1>   
< World, 2>

The output of the second map, sorted:

< Goodbye, 1>   
< Hadoop, 1>

< Hadoop, 1>   
< Hello, 1>

and then combined :  
< Goodbye, 1>   
< Hadoop, 2>   
< Hello, 1>

Shuffle :

Input from the 2 Mappper-combiner

< Bye, 1>   
< Hello, 1>   
< World, 2>

< Goodbye, 1>   
< Hadoop, 2>   
< Hello, 1>

Output :

<Bye,(1)>

<GoodBye,(1)>

<Hello,(1,1)>

<Hadoop,(2)>

<World,(2)>

Reducer :

Input to the reducer

<Bye,(1)>

<GoodBye,(1)>

<Hello,(1,1)>

<Hadoop,(2)>

<World,(2)>

Output of the reducer =

< Bye, 1>   
< Goodbye, 1>   
< Hadoop, 2>   
< Hello, **2**>   
< World, 2>

Typically the compute nodes and the storage nodes are the same. The MapReduce framework consists of a **single master JobTracker** and **one slave TaskTracker per cluster-node**. The master is responsible for scheduling the jobs' component tasks on the slaves, monitoring them and re-executing the failed tasks. The slaves execute the tasks as directed by the master.