Big Data

Rathinaraja Jeyaraj

Don’t try read line by line, look into words highlighted with colours

red	- points to observe now
green	- points to remember always

What is Big data?

any data that is beyond storage and computing (CPU + algorithm) capability of a machine is called big data. Ex:

- 5 GB high definition video is big data for smartphones.
- Rendering 10 GB 3D graphics data is big data for commercial laptop.

Big data refers to a collection of datasets that are huge or flow large enough or with diverse types of data or any of these combinations that outpaces our traditional storage (RDBMS) and computing capability (huge servers, algorithms) to store, process, analyse and understand with cost effective way.

- Volume is one of the factors that choke system capability.
- volume, velocity and variety need not be combined together to say a dataset is big data.
- Any one of the factors (volume or velocity or variety) is enough to say a field is facing big data problems.

- “big data” is not only just emerged from storage capacity point of view, but also from “computational capability and algorithm ability” of a machine.
- hardware processing capability and algorithm’s ability decide how much quantity of data a machine can process in specified amount of time.
- some definitions focus on what data is, while others focus on what data does.

- International Digital Corporation (IDC)
  - estimates every year data created is doubled.
  - estimates over 2.7 ZB of data exist in digital universe today.
  - predicts global IP traffic will be 40 ZB in 2020.

- In today’s world digital data, 90% was created in last couple of years, in which 95% of data is in the semi and unstructured form. Merely, 5% belongs structured data.

History of data generation

before 1980's	-	devices were generating data.
1980 - 2000	-	employees generated data as an end user.
from 2000	-	people started contributing data through social applications, e-mails, blogs…
after 2005	-	every software, application started generating data, especially on internet.

- new technologies, devices, and communication means like social media trigger human to generate data rapidly.
- The amount of data produced from the beginning of digital age till 2003 was 5 billion GB.
- The same amount was created in every two days in 2011, and every ten minutes in 2013.

- Data size was the first, and at times the only dimension that indicates big data.

Big (huge) + data (volume + velocity + variety) → huge in data Volume + huge in data Velocity + huge in data

Big data sources

- Traditional data were just documents, logs and transaction files...
- Nowadays, data are in different form like audio, video, image, 3D, spatial, temporal…

Major domains of big data

- business (E-commerce)
- public administration
- scientific research

Big data in business (E-commerce)

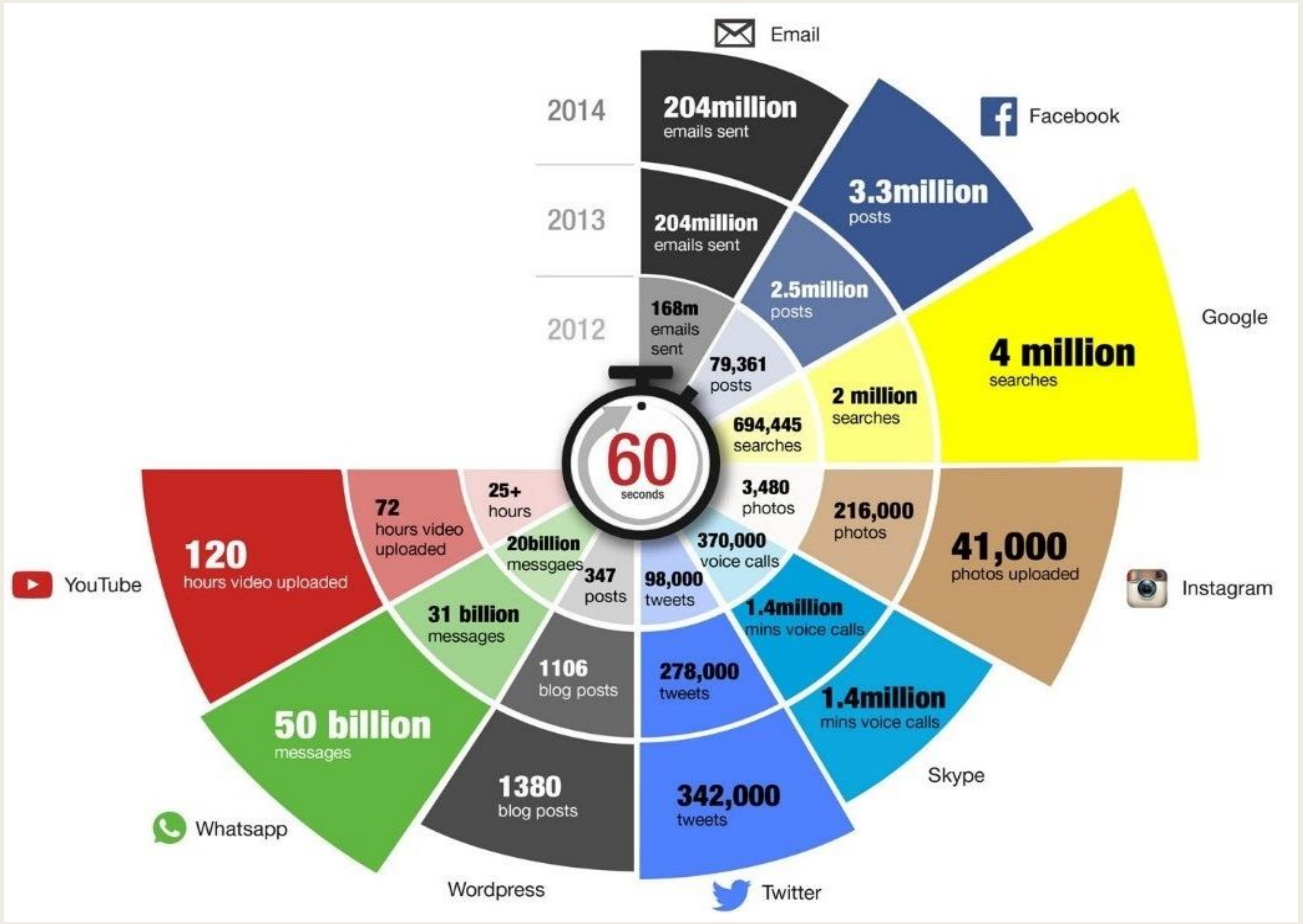
- Volume of business data worldwide doubles every 1.2 years.
- E-com companies such as flipcart, eBay… generate millions of transactions per day.
  - 300 million transactions/day are happening in 6000 wal-mart stores worldwide.
  - 7.5M RFID entries were done in 2005.
  - 30B RFID entries were done in 2012.
  - 100B RFID entries were done in 2015.

- Multimedia and graphics companies face big data problem. Ex: for avatar movie 3D rendering, they required over 1 PB of local storage.
- Credit card fraud detection system manages over 2 billion valid accounts around the world. IDC estimates, over 450 billion transactions will happen per day after 2020.
- NewYork Stock Exchange generate over 1 TB of trade data per day.
- finance, manufacturing, marketing, insurance, retail, pharmaceutical …

Internet based companies

hard to find any activity that doesn't generate data on internet. We are monitored on internet.

- Google indexes trillion pages/day and processes 20 PB/day.
- FB users generate .5 PB/day by exchanging over 3 billion pieces of content.
- Twitter users generate 15 TB/day.
- Image archives (instagram).
- video archives (youtube) uploads 120 hours of new video every second.
- Email, message apps (WatsApp, Hike…).
- Application/system/web logs.
- internet text documents.


Email
Facebook
Google
Instagram
Skype
Twitter
Wordpress
Whatsapp
YouTube
2014
2013
2012
204million emails sent
204million emails sent
168m emails sent
3.3million posts
2.5million posts
79,361 posts
4 million searches
2 million searches
694,445 searches
41,000 photos uploaded
216,000 photos
3,480 photos
1.4million mins voice calls
1.4million mins voice calls
370,000 voice calls
342,000 tweets
278,000 tweets
98,000 tweets
1380 blog posts
1106 blog posts
347 posts
50 billion messages
31 billion messages
20billion messgaes
120 hours video uploaded
72 hours video uploaded
25+ hours
60 seconds

Big data in public administration

- Aadhar is a unique identification number and a large biometric database (over PB) recorded with every person’s retina, thumb impression, photo...
- People in each level need different public services. Ex: kids and teen-agers need more education, elders require higher level health care services…
- Government should provide a higher level of public services with significant budgetary constraints.
- 235 TB of data has been collected by US library of congress in 2011.

Big data in scientific research

Most of the scientific fields have already become highly data driven with development of computer science.

- astronomy, atmospheric science, earth science, meteorology
- social computing, bio-informatics, bio-chemistry, medicine, genomics
- oceanography, geology, sociology…

All these fields produce large volume of data with various types and different speed from different sources.

Several scientific field’s that are overloaded with data are:

- National Science Foundation (NSF) embedded fiber-optic cable over 1000 km’s on sea floor, connecting 1000’s of biological sensors to observe ocean.
- Particle accelerator experiment device such as Large Hadron Collider (LHC) produce 60 TB/day.
- National Climatic Data Centre (NCDC) generates over 100 TB/day.
- Large Synoptic Survey Telescope (LSST) - a large digital camera that records 30 trillion bytes of images/day.
- Sloan Digital Sky Survey (SDSS) produces 200 GB images/day.

Other sources

- IoT and sensor networks .
- CCTV for military surveillance, animal tracking...
- Call logs in customer service center.
- Companies shortlisting CVs from millions of applications.
- Aircrafts generate 10 TB black box data every 30 mins.
- Power grid data.

Why should we store and process big data?

We started accumulating data as storage device price is cheaper. We need historical data

- for decision making.
- to learn from past data to adapt changes and reacting swiftly for future.

The potential of big data is in its ability to solve problems and provide new opportunities.

Ex: one of the FB revenue sources is publishing ads on the page you like, share, comment… They allow E-com companies to access your data for money to publish ads.

Ultimately, processing and extracting insight from big data should lead to

- increase productivity in business.
- improve operational efficiency, reduce risks and make strategic decision in management.
- ease scientific research.

big data applications

- Business: recommender system (people who like this product may also like another product in online shopping, FB's friends request…), sentiment analysis…
- Social network analysis: detecting online communities, predicting market trends, ads targeting…
- E-commerce and retail industries: Marketing, product recommendation…
- Banking and Finance: stock market analysis, risk management, fraud detection…
- Transportation: logistic optimization, real-time traffic flow optimization…

- Healthcare: medical record analytics, genomics (sequencing and decoding human genes took 10 years to process), patient monitoring…
- Telecommunication: call record processing, thread detection…
- Entertainment: Multimedia and graphics (avatar movie 3D rendering required over 1 PB of local storage)...
- Government to forecast events and taking proactive measures against spread of disease, natural disaster…

Big data characteristics

Any amount of data is big data when storage, computational, and algorithm ability fail to process and extract meaningful insight from dataset.

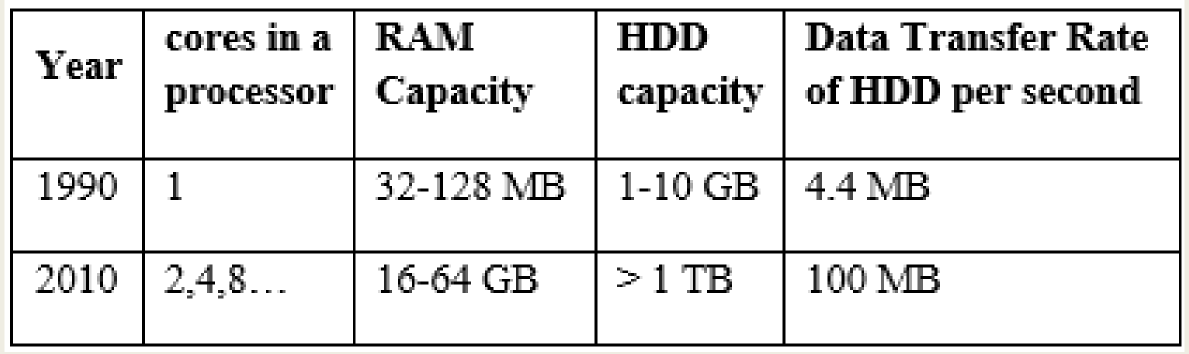
Following are the indicators to mention big data.

- Volume
- Velocity
- Variety
- Veracity
- Value
- Variability

volume

more data more accurate decisions

- Data size that is beyond the storage capacity of a machine.
- Big data should not be discarded, because storage cost is cheap.
- Data that is not processed today may be worth enough when processed tomorrow.
- Different users have different demands.
- Relational models (RDBMS, DWH) fail to store and manage huge data in tables.
- It takes over a day to process few TB data as disk transfer rate is just 100 MB in enterprise level servers.
- There is a statement "data growth exceeds moore's law". What does it mean? CPU heavy IO poor problem



Year	cores in a processor	RAM Capacity	HDD capacity	Data Transfer Rate of HDD per second
1990	1	32-128 MB	1-10 GB	4.4 MB
2010	2,4,8…	16-64 GB	> 1 TB	100 MB

Data size measurement

1 Bit	- a binary digit. either 0 or 1
1 Byte	- 8 bits
1024 B	- 1Kilo byte
1024 KB	- 1 Mega byte
1024 MB	- 1 Giga byte
1024 GB	- 1 Tera byte
1024 TB	- 1 Peta byte
1024 PB	- 1 Exa byte
1024 EB	- 1 Zetta byte
1024 ZB	- 1 Yotta byte
1024 YB	- 1 Bronto byte
1024 BB	- 1 GeoP byte

Velocity

more faster more value

- Rate of incoming data to a machine for processing is called velocity - realtime stream processing
- System should provide fresh, low latency results in real-time and give space for new data.
- Relational models (RDBMS, DWH) is not suitable for data velocity as needs to index data before accessing.
- Real-time applications:

    threat detection in tele-communication

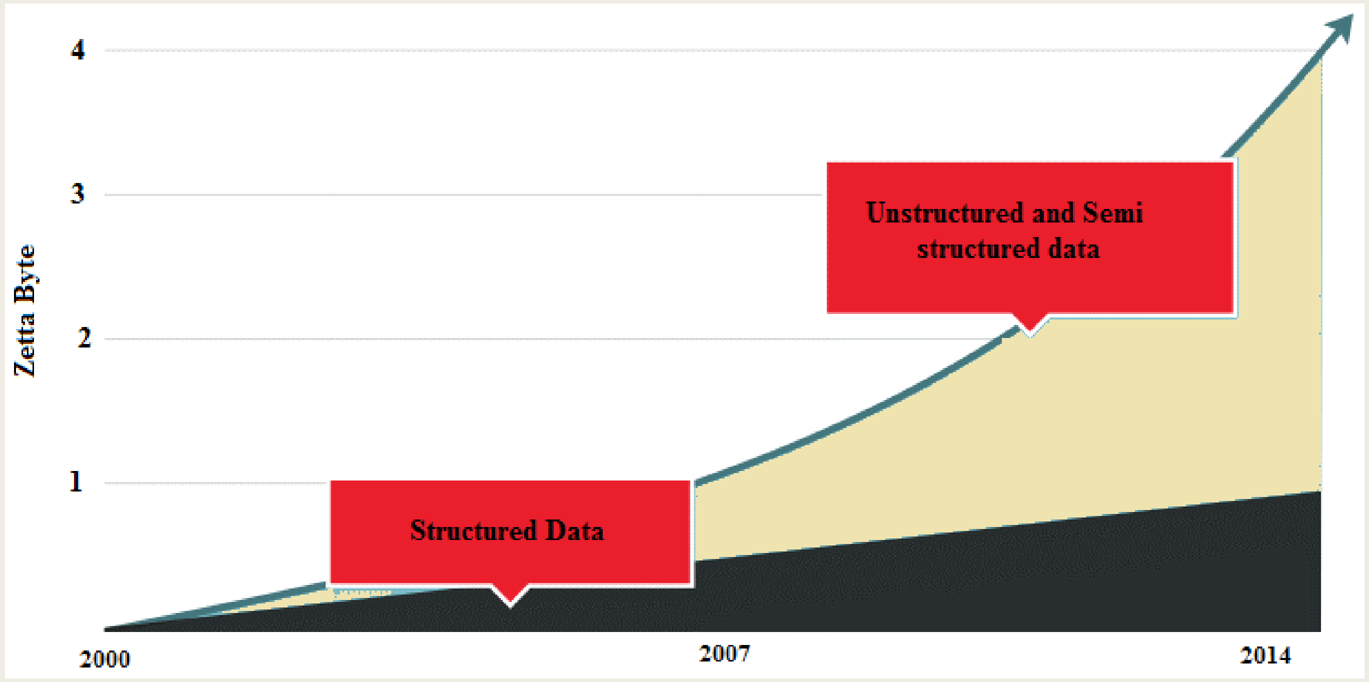
    fraud detection in banking

    recommender system in social and e-commerce apps…

Variety

one tool processing different data

- Structured data in relational models (RDBMS, DWH) grows linearly in banking and other business sections.
- Unstructured and semi-structured data grow exponentially due to internet based applications and IoT.
- Every year unstructured data is doubled.
- Data processing tool must have capability to process any type of data.



Structured data:

- create table (schema) before you accumulate and manage data in an organized way.
- Schema changing after huge data accumulation is very costly in relational models (RDBMS, DWH).

Unstructured data:

- denotes data organization with dynamic schema or no schema. Ex: FB user uploads image, video, audio, text (chat, status, comment)…
- such heterogeneous data cannot be put in relational models (RDBMS, DWH) as schema modification is costly for varying size data.
- Answer is NoSQL databases.

Semi-structured data:

- it is not as structured as relational databases.
- web pages have little structure with tags, but no restriction with the data between the tags.
- log data generated by machines/software/applications.

Example: Every person has one or more E-mail, Facebook, twitter, blog… accounts. Minimum 10 MB of log data per day is generated while accessing those sites.

This log data is processed by service providers to track user behaviour for many purposes like product recommendation, marketing campaign… Imagine if 1 million users generate 10 MB of log data in a day, then it is roughly 10^6 x 10 MB (over 10 TB).

Similarly, 20+ million web pages generate (20 KB x 20×10^6) 400+ TB log data per day. Enterprise server's maximum capacity to read from disk is 100 MB/sec. Therefore, it takes few days to just read web log data.

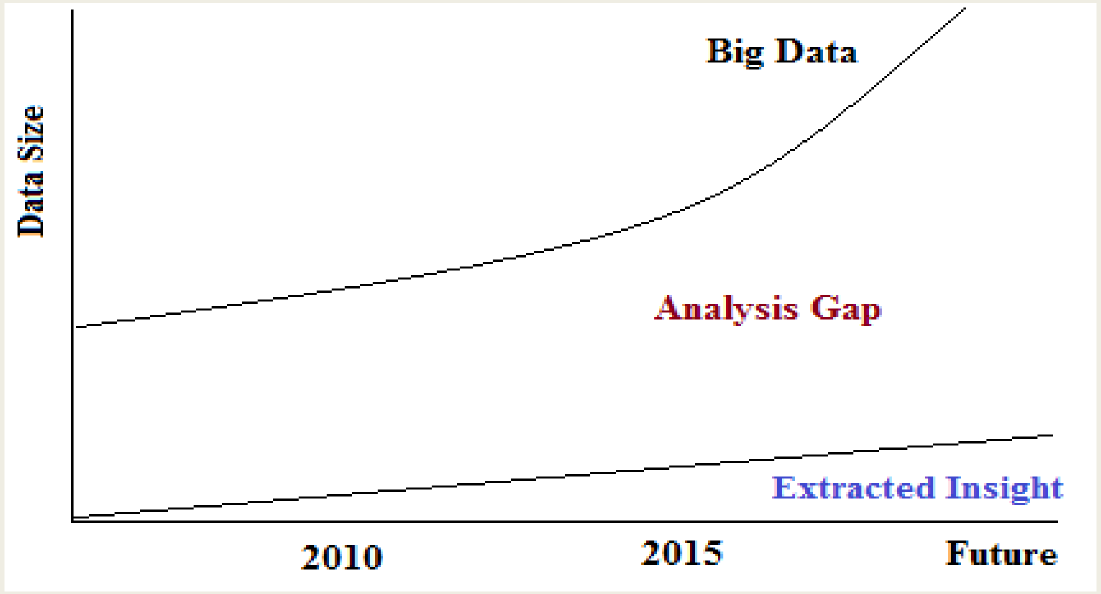
Different data sources:

- Structured data – banking, finance, business sections…
- Semi-structured data – log data from hardware/software/applications, emails, web pages with XML/HTML…
- Un-structured data – audio, video, text documents…

value

big data beats better algorithms

- It depends on the ability of algorithm to extract potential insight from any amount of data.
- Relevant insight extracted from big data could be very less.



Veracity and variability

Veracity: processing data taken from public sources such as social networks may not be accurate most of the time, because authenticity of users (anybody can post any data) is not reliable on internet.

Variability: Data flow rate is not constant in streaming data.

A company may face any one of the Vs or combination of any Vs. However, 'first three-V' exists in every firm.

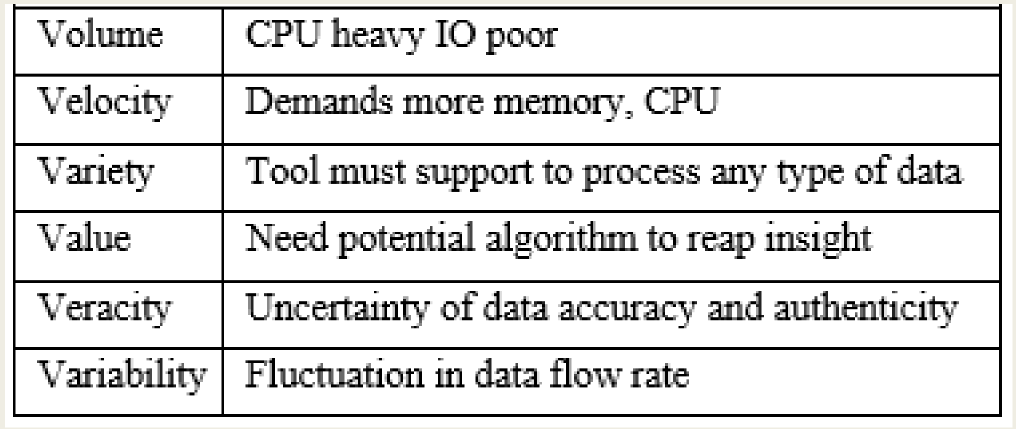
No universal benchmarks for volume, variety, and velocity to measure big data.

The defining limits depend upon the firm and these limits evolve over time.

These Vs may be dependent of each other. If one V changes, another V may get changed.

Different V’s in short

- Volume : indicates data size.
- Velocity : denotes arrival rate of continuous data to a processing machine.
- Variety : talks about data heterogeneity. (structured, unstructured, semi-structured)
- Veracity : discusses uncertainty of accuracy and authenticity of data.
- Value : points less value/information/knowledge from huge data.
- Variability : denotes fluctuation in data flow rate.



Volume	CPU heavy IO poor
Velocity	Demands more memory, CPU
Variety	Tool must support to process any type of data
Value	Need potential algorithm to reap insight
Veracity	Uncertainty of data accuracy and authenticity
Variability	Fluctuation in data flow rate

Big data scenario examples

Example 1: an insurance agency collects data about a person from various sources such as social media, bank transaction, web activity… and decides whether to display him an insurance ad while booking travel ticket.

Insurance company considers its competitors price and offer better value for attracting customers. Now, insurance company faces volume (historical data of customers such as bank transactions), variety (data from social apps), velocity (click streaming data)…

Example 2: General Motors (a company for manufacturing, selling… vehicles and spare parts) has its own datacenter facility to monitor running car engine's health. They predict the failure of engine and prepare for replacement before car owner arrives.

It certainly increases customer experience. Moreover, they sell these data to insurance agency for deciding insurance claim according to the speed they drive. Accident by rash drive can't claim for insurance. It faces velocity, and volume.

Every sector faces big data. It is not collecting data world wide and processing.

Finding Big Data on Web

- Avito, kaggle, kdnuggests, h2o.ai, itvarsity, canopy
- books: http://gutenberg.org, http://cbt.gg/18yYJHn
- s3data: http://aws.amazon.com/datasets
- wikipedia (large): http://en.wikipedia.org/wiki/wikipedia:database_download (4TB)
- weather dataset: http://cbt.gg/18yYWu6
- https://archive.ics.uci.edu/ml/index.html
- http://sci2s.ugr.es/keel/index.php
- http://www.nltk.org/

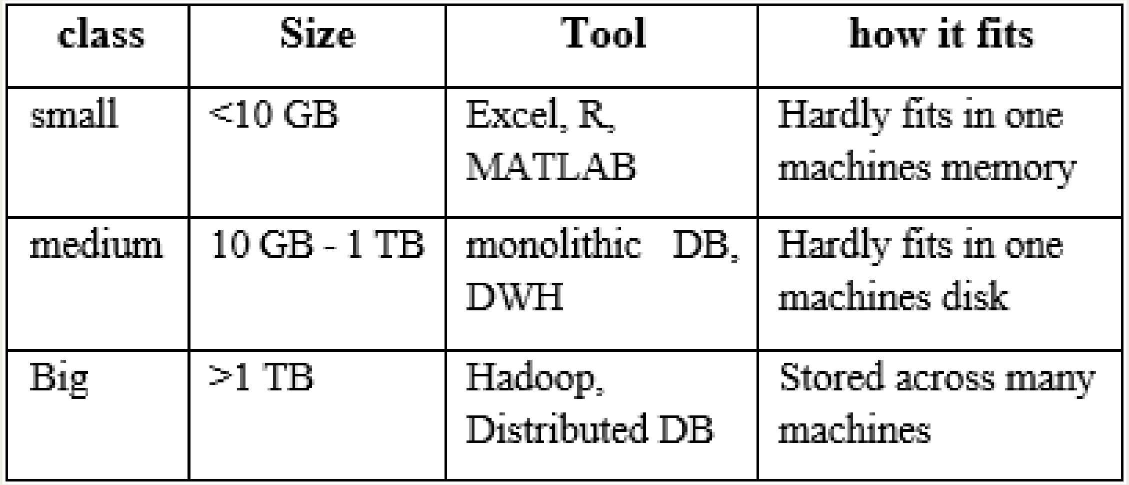
- https://github.com/nltk/nltk
- http://alias-i.com/lingpipe/demos/tutorial/classify/read-me.html
- http://www.datasciencetoolkit.org/developerdocs
- http://web.stanford.edu/class/cs224w/resources.html
- http://hdr.undp.org/en/statistics/data/
- http://www1.ncdc.noaa.gov/pub/data/normals/1981-2010/source-datasets/isdlite-normals.tar.gz
- ftp://ftp.ncdc.noaa.gov/pub/data/uscrn/products/daily01/
- http://wadam-data.dis.uniroma1.it/
- http://eoddata.com/

Scientific paradigm

The emergence of science over centuries is as follows.

- Empirical Science - 1000 years ago, proof was based on experience and evidence verifiable rather than pure theory or logic.
- Theoretical Science - in the last century, proof was theoretically derived (Newton law, Kepler law…) than conducting experiment for many complex problems as creating evidence was difficult. But, it was also infeasible deriving thousands of pages.
- Computational Science - solving problems using computers by constructing a mathematical model is called computational science. It is application of computers to problems that deal with calculation intensive tasks (which are not humanly possible to calculate in short time).
- Data Science - deals with data intensive (huge data) computing.

Data handling tool/framework



class	Size	Tool	how it fits
small	<10 GB	Excel, R, MATLAB	Hardly fits in one machines memory
medium	10 GB - 1 TB	monolithic DB, DWH	Hardly fits in one machines disk
Big	>1 TB	Hadoop, Distributed DB	Stored across many machines

Big data systems

The potential of big data analytics is in its ability to solve business problems and provide new business opportunities by predicting trends.

In E-commerce, ads targeting, collaborative filtering (recommendation system), sentiment analysis, market campaign… are some of the use cases that require to process big data to stay upright in businesses and to increase revenue.

There are two classes of big data systems:

- Operational big data system (Real-time response from big data)
- Analytical or Decision support big data system (Batch processing)

Operational big data system

Transaction: A transaction is set of co-ordinated operations to be performed in sequence. Ex: money transfer.

Transactional database: databases that perform transactions for day to day activities are called transactional databases.

These databases are highly structured and heavily used in banking, finance and other business applications. Ex: RDBMS.

Operational database: databases that don’t support transactions, but still perform day to day activities (such as insert, update, delete) is called operational databases.

It is highly used in web based applications such as Facebook, WhatsApp… Ex: NoSQL databases such as Hbase, Cassandra (Facebook), MongoDB, BigTable (Google)…

- NoSQL databases are only operational databases and not transactional databases. Because, transactional activities may require synchronization to ensure consistency that limit scalability of distributed system. MongoDB tries to support both transactional and operational abilities.

- Both transactional and operational databases provide real-time response to the user. However, transactional databases are write consistent and operational databases are read consistent.

- Some NoSQL big data systems provide few analytical functions to derive insights with minimal coding effort and without the need for data scientists and additional infrastructure.

- NoSQL big data systems are designed to take advantage of cloud computing to adapt massive scalable computing and storage inexpensively and efficiently. This makes operational big data workloads much easier to manage.

Analytical or Decision support big data system

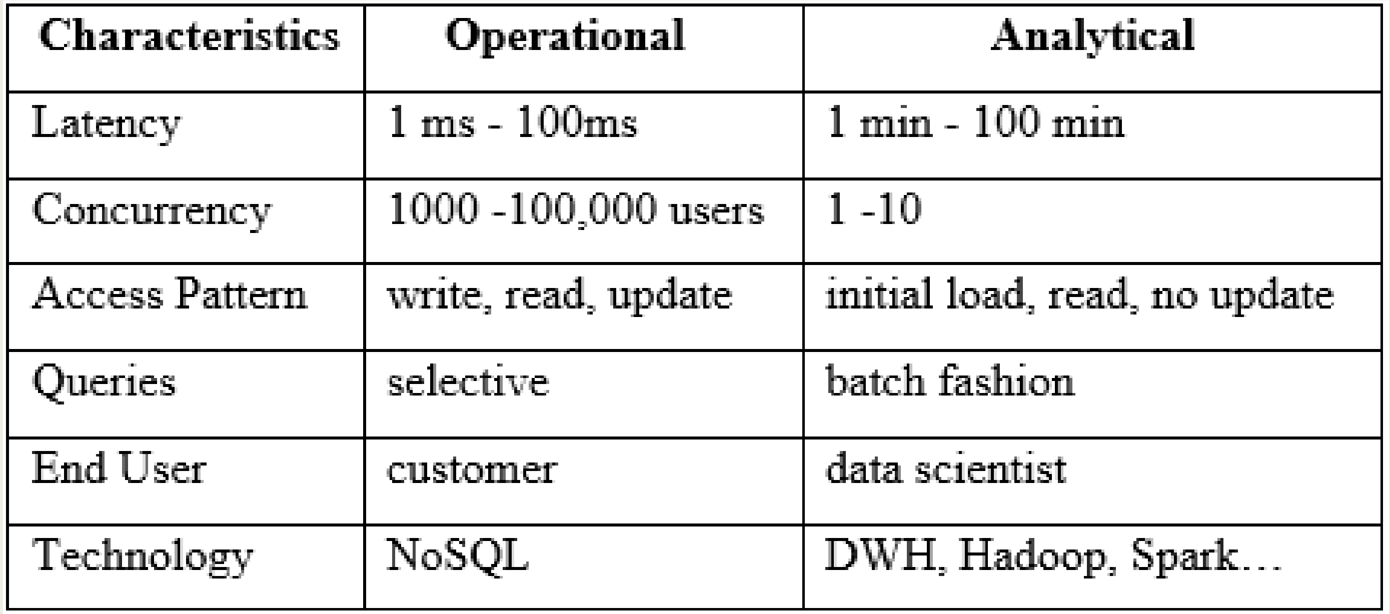
- Big data analytics refers to sequence of steps (capture, store, manage, process, analyze, understand and visualize/interpret) carried out to discover unknown hidden pattern/trends/relationship/associations and other useful information from big data for decision making process with cost effective way.

- Processing historical huge amount of data using data mining and machine learning algorithms to extract insight and other useful information for decision making is called analytical big data systems. It is also called decision support system (DSS).

- It highly relies on batch processing that takes minutes to hours to give response. These include systems like
- Data WareHouse (DWH) stores historical structured data and provides pre-defined queries (OLAP).

- Distributed storage such as HDFS, S3, azure blob, swift… to store semi and unstructured data with no pre-defined queries. You have to write your own algorithms (called adhoc-algorithm) to process huge data.

Operational vs Analytical big data systems



Characteristics	Operational	Analytical
Latency	1 ms - 100ms	1 min - 100 min
Concurrency	1000 -100,000 users	1 -10
Access Pattern	write, read, update	initial load, read, no update
Queries	selective	batch fashion
End User	customer	data scientist
Technology	NoSQL	DWH, Hadoop, Spark…

Platform for big data analytics

What do we do when there is too much of data/computation to process? Scale the system.

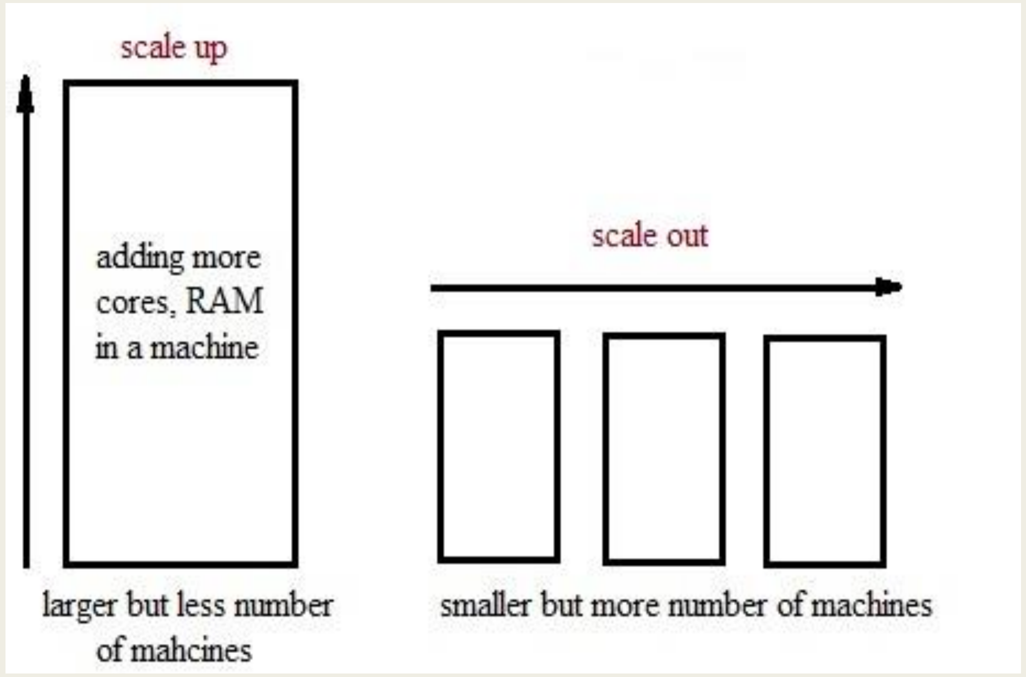
- Ability of a system to adapt more resources (CPU, memory, storage…) to tackle increasing amount of data/computation.
- scaling computer systems and algorithms are being the great challenges to tackle increased data.

Two types in scaling the system : scale up and scale out.

Horizontal scaling (Scale out)

Resizing the cluster by adding multiple computers that work together as a single logical unit.

- We distribute computation/data across many servers to process in parallel.
- It can be scaled out on the fly by adding servers while cluster is up and running.
- It is suitable for offline (batch) processing.
- Scale out is more challenging to develop software that runs in distributed environment to handle fault tolerance.
- hardware costs, and software licensing are cheaper.
- Increases performance linearly.
- Ex: P to P network, Hadoop, Spark...


scale up
adding more cores, RAM in a machine
larger but less number of mahcines
scale out
smaller but more number of machines

Vertical scaling (Scale up)

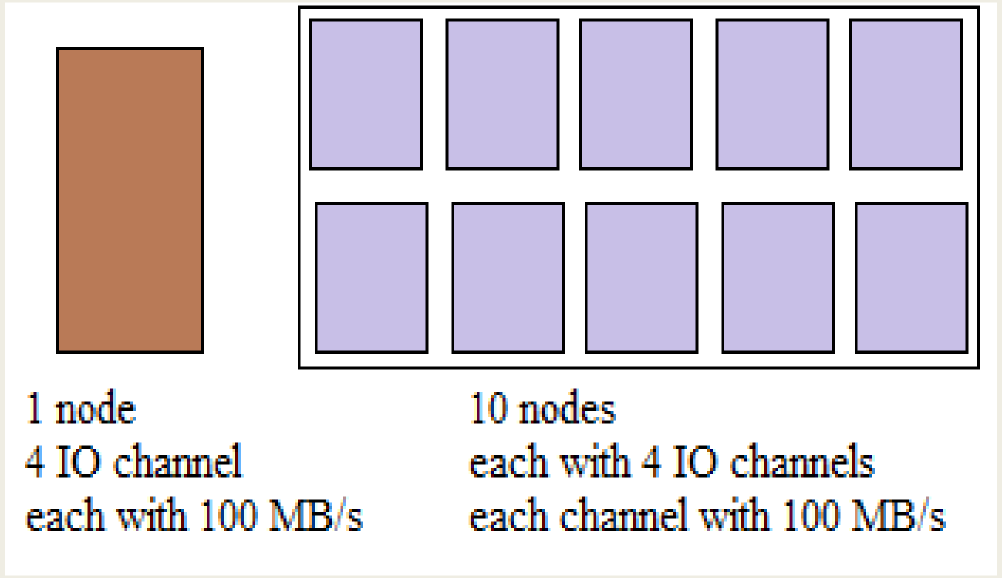
Resizing a computer by adding more processors and memory.

- scaling up reached its limit imposed by hardware.
- It is suitable for real-time processing.
- Adding resources to a server cannot be done on the fly, so, it needs down time.
- Scale up is more expensive than scaling out.

- Scale up leads to CPU heavy I/O poor problem. Therefore, there is no use of increasing resources of a machine beyond a point for big data processing (but fruitful for compute-intensive tasks).
- That is the reason supercomputers and GPU are not preferred for big data processing.
- Moreover, failure of anything in such systems is very expensive to treat (needs down time) and not affordable for most of the small-scale companies.
- Ex: GPU, FPGA.

What is for big data analytics? Scale out or scale up

To read 1 TB data from HDD



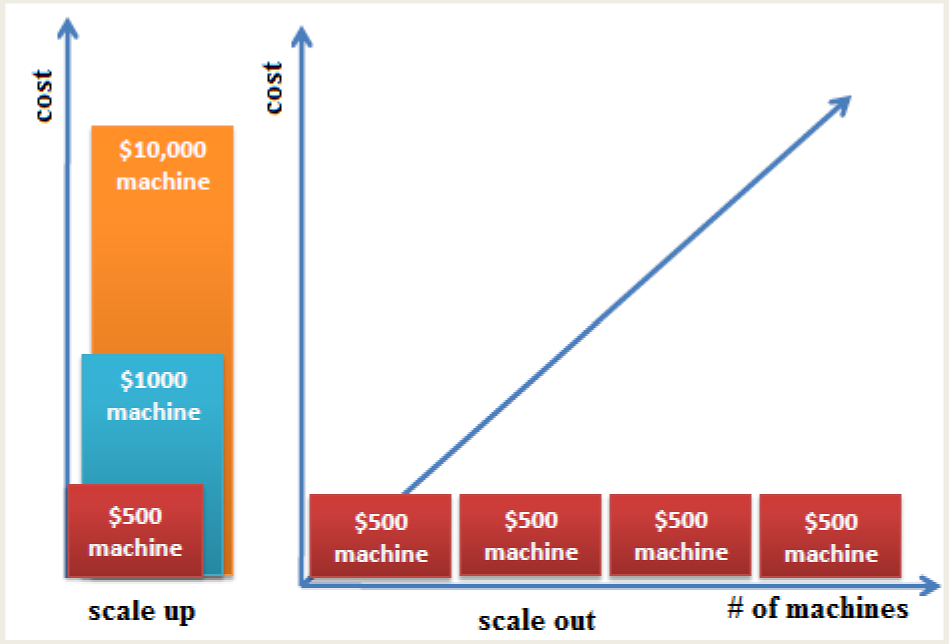
Using single machine = $\frac{1\,\mathrm{TB}}{4*100\,\mathrm{MB/sec}} = \frac{2^{40}}{400*2^{20}\,/\,\mathrm{sec}} = \frac{2^{20}}{400}\,\mathrm{sec} = 2621.44\,\mathrm{sec} = 43.69\,\mathrm{min}$

Using cluster of 10 machines each storing part of big data, say 102.4 GB =

$$\frac{1\,\mathrm{TB}}{10*4*100\,\mathrm{MB/sec}} = \frac{2^{40}}{4000*2^{20}\,/\,\mathrm{sec}} = \frac{2^{20}}{4000}\,\mathrm{sec} = 262.144\,\mathrm{sec}s = 4.369\,\mathrm{min}$$

How to choose a platform?

- Crucial to choose a right platform for right problems: decision depends on
  - how fast you want the result given huge size of dataset.
  - whether you will need more data processing capability (scalability) in future.
- Scale up is more expensive than scaling out.



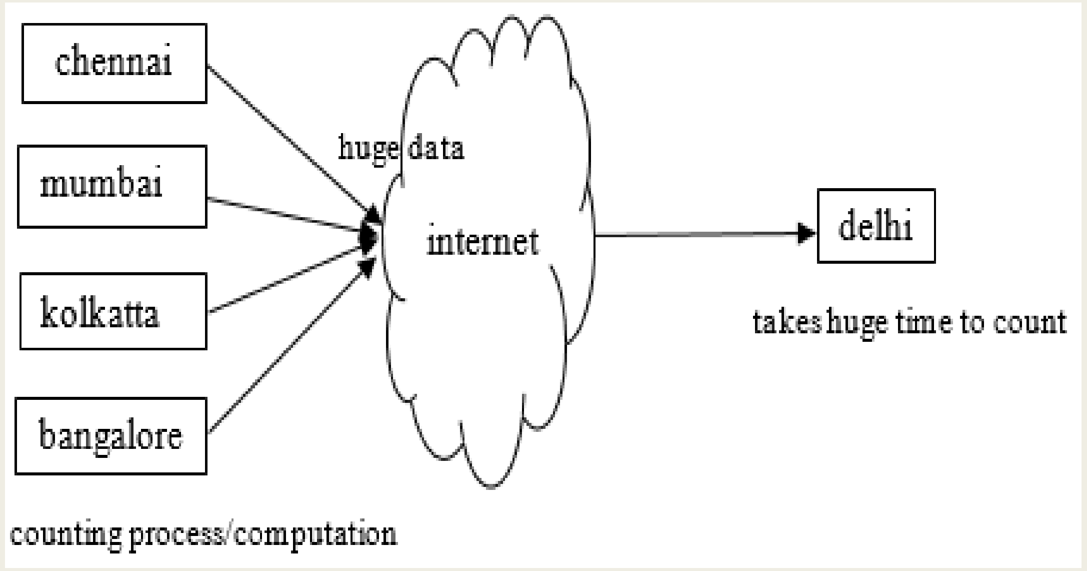
Data-center

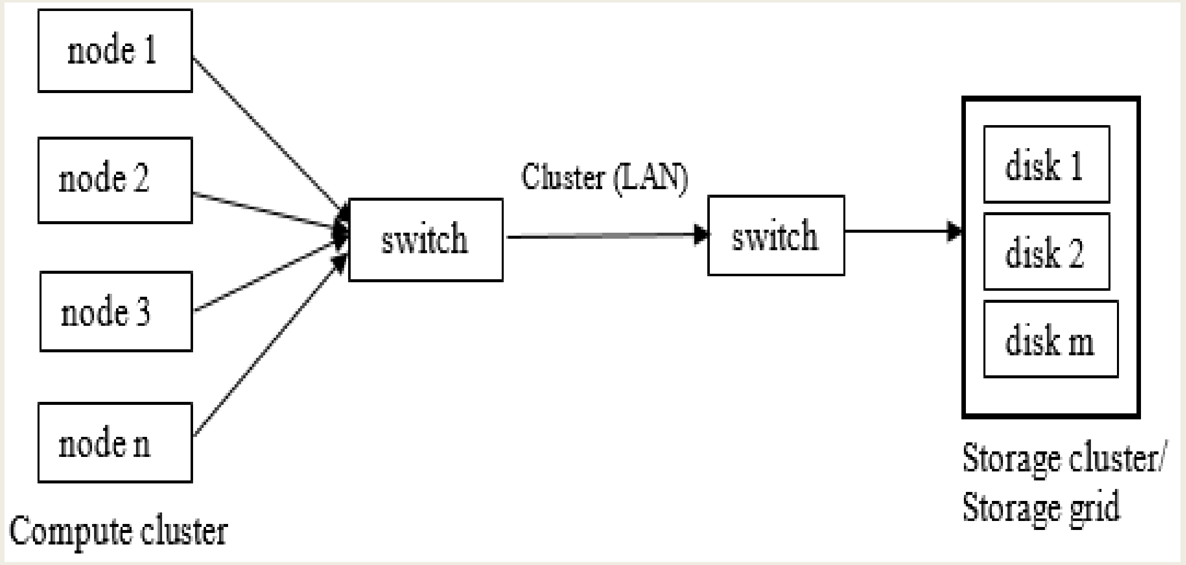
Server cluster

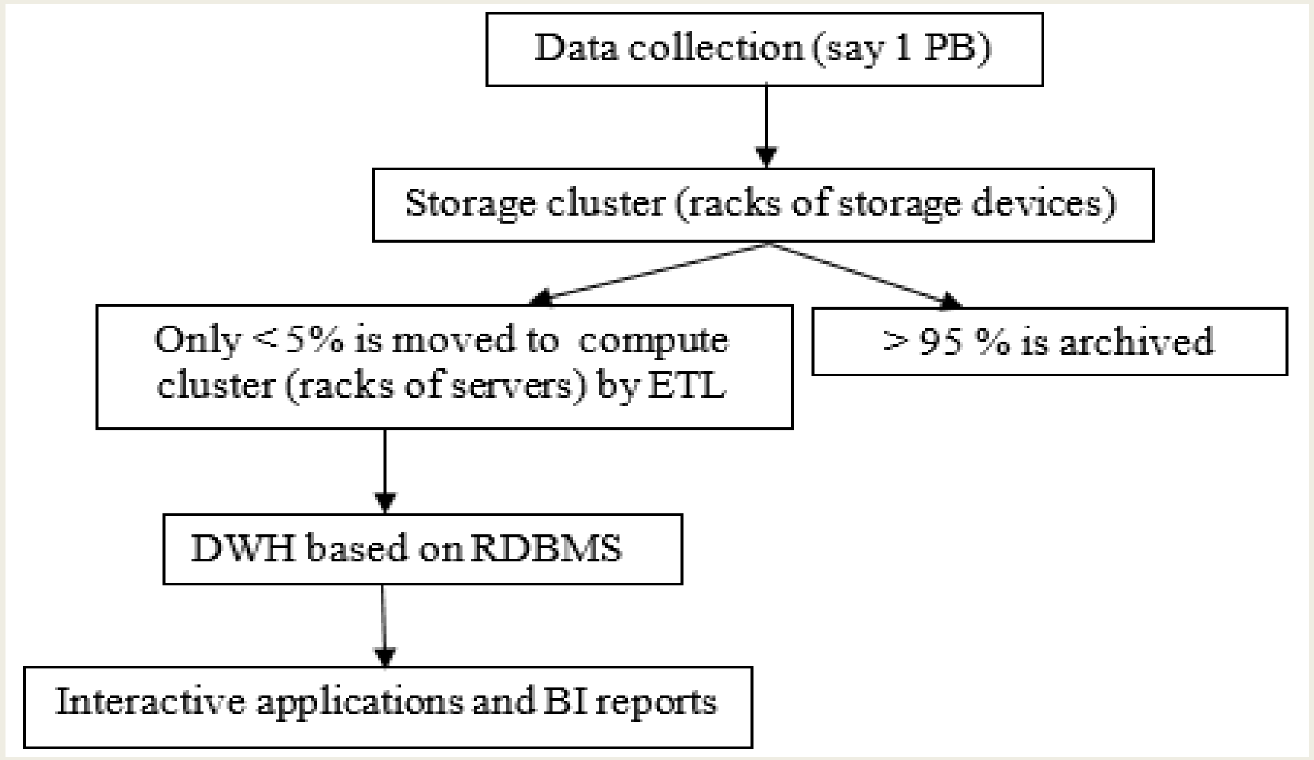
storage cluster

Existing scale out data analytics framework

move data to node which runs program



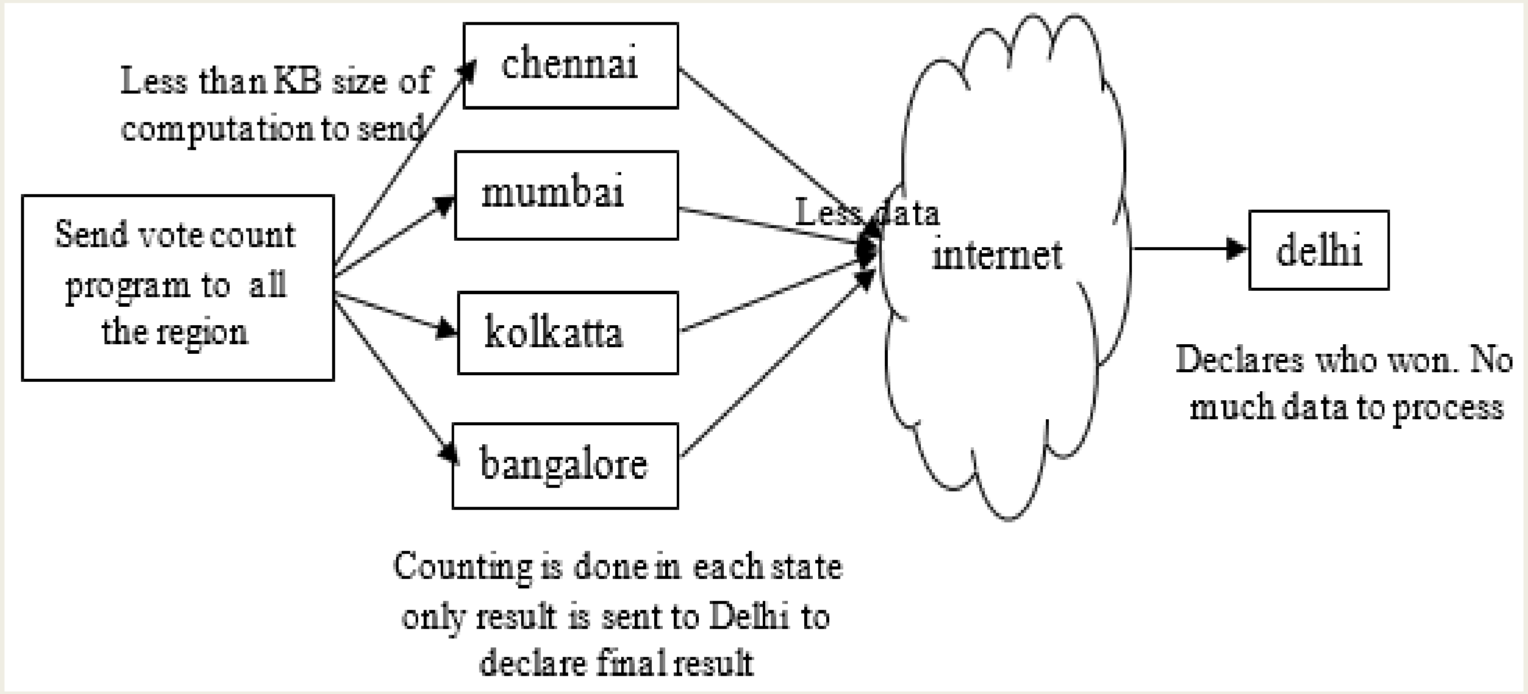


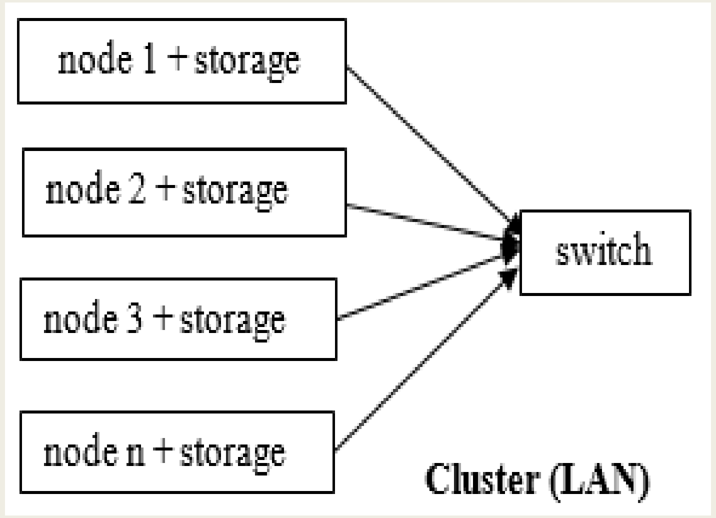

Data collection (say 1 PB)
Storage cluster (racks of storage devices)
Only < 5% is moved to compute cluster (racks of servers) by ETL
> 95 % is archived
DWH based on RDBMS
Interactive applications and BI reports

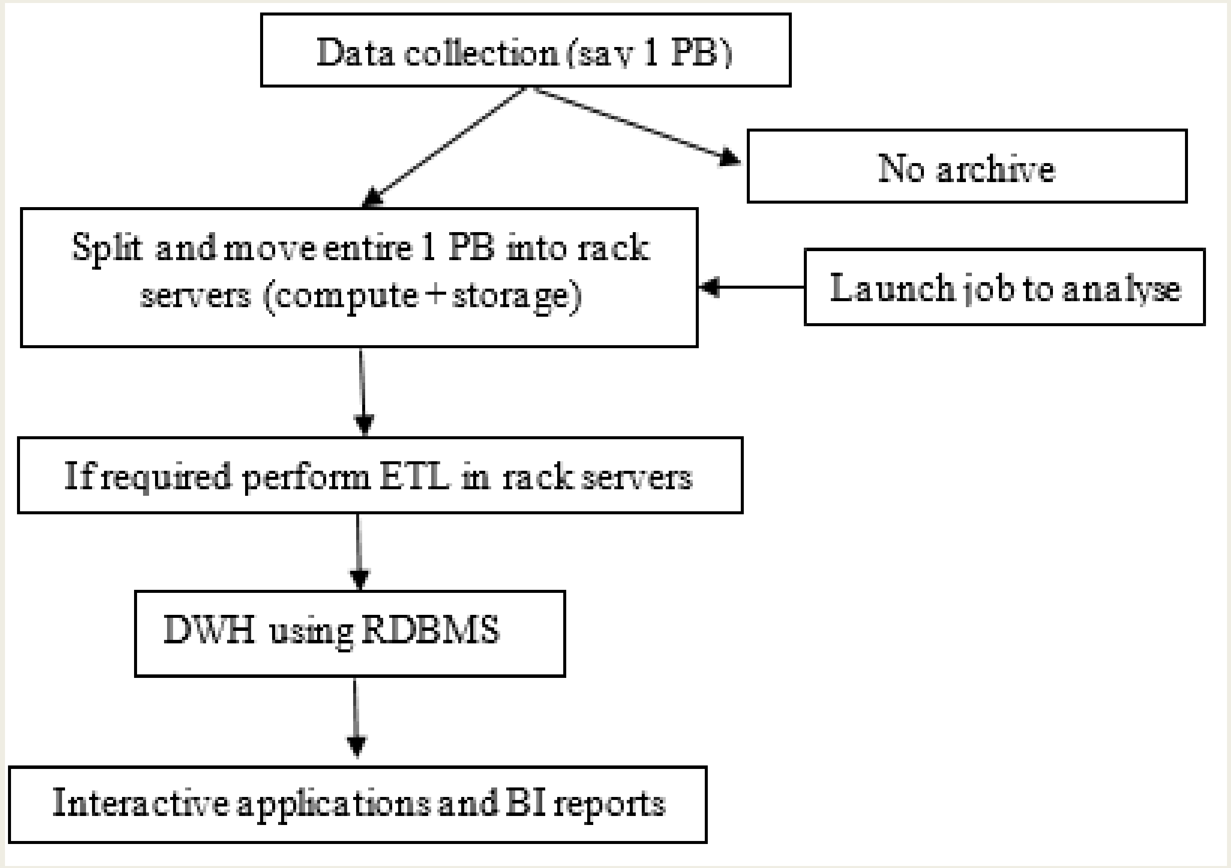
- Archived data may have more meaningful insights, but not processed yet.
- Relational DWH tools to build and manage structured DWH: Informatica, Power center, Teradata, Exadata, Syncsort, Cognos….

Advanced data processing framework

move program/computation to node having data


Less than KB size of computation to send
chennai
mumbai
kolkatta
bangalore
Send vote count program to all the region
Less data
internet
delhi
Declares who won. No much data to process
Counting is done in each state only result is sent to Delhi to declare final result


node 1 + storage
node 2 + storage
node 3 + storage
node n + storage
switch
Cluster (LAN)


Data collection (say 1 PB)
No archive
Split and move entire 1 PB into rack servers (compute + storage)
Launch job to analyse
If required perform ETL in rack servers
DWH using RDBMS
Interactive applications and BI reports

Another example:

you have personal data (say 10 TB) that can't be managed in your laptop or desktop. Therefore, you approach a data-center (cloud) and upload your data.

To process them, if you bring all those data to your machine, then it will take lot of public bandwidth and will take huge time to process with your machine.

Therefore, create a program and send to the data-center where you have stored data and get the result back.

Advantages advanced data processing framework

- data locality - code/program goes to data.
- Scalability (almost linear) - as you increase the cluster size, performance increases.
- Low cost hardware (commodity machine).

Challenges in horizontal framework

- powerful only when there is no communication, blocking, and synchronization among processes.
- Problems in cluster computing that must be addressed.
  - Moving data from storage cluster to computation cluster is costly.
  - Computers fail every day. So, server failure is expected.
  - Data is corrupted or lost.
  - Computations are disrupted.
  - The number of nodes in a cluster may not be constant.
  - Nodes can be heterogeneous.
- Handling above problems using MPI for scale out architecture is possible. But, it is application programmer responsibility to distribute data, program, handling failure, managing cluster, networking…
- difficult to build reliability into each application/job.
- reduces application development productivity.
- needs a common infrastructure that handles above responsibilities.

Requirements of distributed system for advanced data processing framework

- Distributed storage and distributed computing (data locality - code/program goes to data).
- Low cost hardware (commodity machine).
- Partial failure: failure of a node should result in a graceful degradation of application performance, but not the failure of the entire system.
- Scalability: adding nodes should result linear increase in performance.
- Fault tolerance: ability to recover data, task, node from failure.
- Consistency: node failure during execution of a job should not affect the outcome of the job.
- Node recovery: if a failure node is repaired, then it should be able to rejoin the cluster.

Existing big data analytics framework drawbacks

- For decision support system, we used DWH to store historical data about a subject and processed using OLAP.
- DWH could not store unstructured data, lacked scalability, sampled or pre-processed data.

1. Consider you have 1 TB HDD in your machine.

Case 1: Can you store 300 GB of data? Yes. Can you store 700 GB of data? Yes.

But, processing (read from disk, perform analysis) 500 GB of data by single machine may take more than a day.

Therefore, even huge servers and GPU are not cost effective to process huge data.

Case 2: Can you store 2 TB of data? No. it is beyond the storage capacity (1 TB) of a machine.

Solution: use cluster of machines.

Divide huge data into chunks, store in multiple machines in the cluster, process data in parallel with same program, and combine the result finally.

2. Previously, peer-to-peer network (storage-compute clusters) with MPI was used to process huge data.

Computation was processor bound (move data to the node where program is running) and collect back the result.

It was highly network intensive and gave its worst for increasing data volume.

Solution: store data in the local storage of server in compute cluster (no storage cluster).

Move computation (program) to node which has required data and process data there itself without moving on network (data locality/data gravity).

helps in horizontal scalability, because, program size is very less (few KB) to transfer than data.

3. Cheap nodes fail, especially if you have many. So, it is hard to ensure fault tolerance.

node/networking failure, distributing program/data, data/program failure… should be taken care by programmer self for each application separately.

Solution: need a common framework that handles all these responsibilities of distributed system and let programmers only to write applications.

Hadoop is an answer to solve those three problems

Hadoop

- an open-source software that allows to store and process big data in a distributed environment across clusters commodity computers using simple programming model.
- designed to scale out from single machine to thousands of machines each offering its computation and storage.
- challenge is not storing huge data (storage is cheap), but, it is on processing speed and developing distributed algorithms.

- highly distributed, horizontally scalable, fault tolerant, high throughput, flexible, and cost-effective software solution for big data processing.
- any small-scale companies can use and modify to their requirement.

- Big data solution is just more than just Hadoop. It is deriving insight from huge data using data mining, machine learning algorithms for decision making process.

Hadoop History

- Google was one of the first organizations to face big data problem.
- wanted to download the whole internet and index it to support search queries with MYSQL database.
- indexed 1 million pages in 1998, one billion in 2008, and over trillion pages every day after 2010.
- implemented Google File System (GFS), Google Map Reduce (GMR), and Big Table in C++ for large-scale index data processing.
- Doug Cutting (creator of Apache Lucene for text search) and Mike Caferella implemented Hadoop Distributed File System (HDFS) and MapReduce (MR) in java based on GFS and GMR.
- Scalability to support huge data and huge processing in search engine was the driving factor for Hadoop invention.

- 2001 – Lucene: Text search library
- 2002 – Nutch: open source web search engine in Yahoo. It required hardware of $500000 with monthly running cost $300000 and not scalable. They tried scaling the storage and processing up to 20-600 nodes (Dreadnaught project).
- 2003 – GFS whitepaper
- 2004 – Nutch Distributed File System (NDFS)
- 2004 – GMR whitepaper
- 2005 – Nutch MapReduce (NMR)
- 2006 – NDFS + NMR were renamed as Hadoop (as sub project of Nutch) and separated from Nutch project.
- 2008 – Hadoop was handed over to Apache software foundation and renamed to Apache Hadoop.
- Now – Apache is directing and managing Hadoop and its sub projects offering as open source.

Software release

- Alpha release: application is working, but some functionalities is likely to be missing. A number of known and unknown bugs are likely to surface.
- Beta release: tested internally and by the wider community, has some bugs in completed software, speed/performance issues, and may cause crash or data loss.
- Stable release: entirely tested and ready for use. Its behaviour, functionality, specification, API is considered ‘final’ for that version. Apart from security patches and bug fixes, the software will not change for usually from 1 to many years.
- You must always download stable release for Hadoop installation. Standard release versions are named as “x.y.z”. Ex: Hadoop 1.2.1, Hadoop 2.7.0…
  - x stands for major release (may break compatibilities)
  - y stands for minor release (backward compatible)
  - z stands for point release (backward compatible)
- Hadoop1.x.x versions were behind on API upgrades while latest Hadoop2.x.x runs through architectural upgrades.

OS that supports Hadoop

- Linux (Ubuntu, CentOS, Open Suse, Red Hat…)
- Solaris
- Windows
- Mac

Hadoop distributions

All commercial Hadoop distributions are modifications of original Apache Hadoop.

Lot of vendors provide Hadoop service and support over internet for money.

They also provide virtual machine with pre-installed tools such as HDFS, MapReduce, Pig, Hive, Hbase, Oozie… Such VM is called quick start VM.

You just download that VM and use in your machine. Notable distributors are:

1. Apache Hadoop
2. CloudEra – Cloudera Distribution for Hadoop (CDH)
3. HortornWorks - Hortornworks Data platform (HDP)
4. MapR
5. Greenplum

Hadoop features

Hadoop transforms cluster of commodity hardware into a service that stores and processes PBs of data reliably with cost effective way. Its features are

Highly Distributed

- data and program can be moved around cluster of machines. Hadoop works on distributed file system and distributed computing.

Horizontally Scalable (scale out)

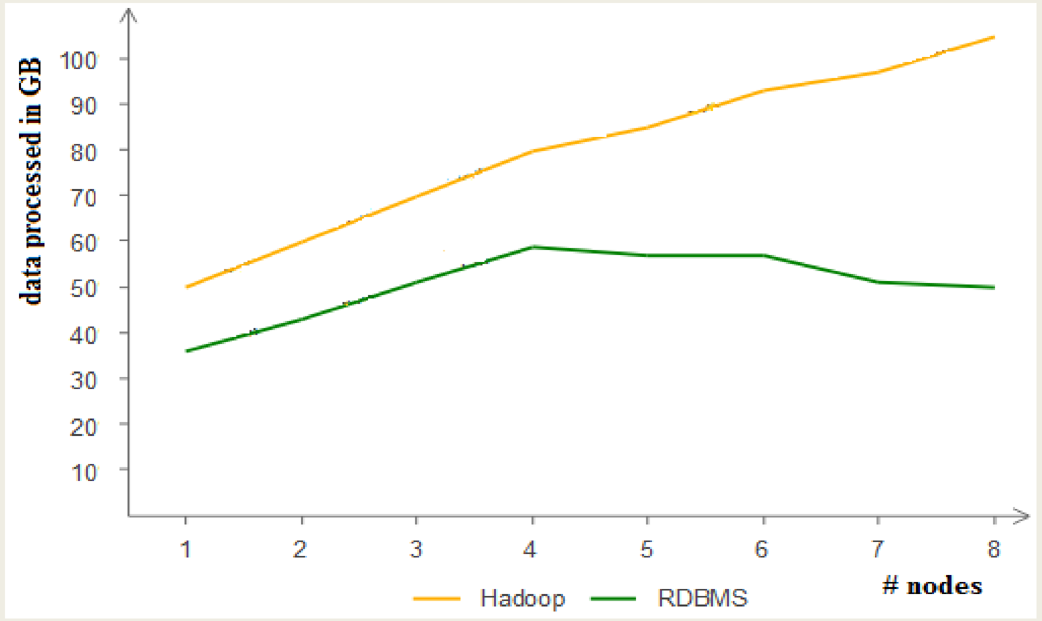
- Primary challenges for data analytics are scalability of cluster, and algorithms. Hadoop supports horizontal scalability for cluster. MapReduce facilitates to write scalable algorithms.
- Theoretically there is no maximum limitation in scaling out.
- You can use the same program written for one system to distribute over 1000’s of machines without the need of re-writing program according to scaling.

Fault Tolerance

- Cheap nodes fail, especially if you have many in the cluster. Mean time between failures (MTBF) for 1 node is 3 years. MTBF for 1000 nodes is about 1 day. So, failure of nodes in a cluster is more likely. Hence, data and computation loss are obvious.
- Fault Tolerance is the ability of a system to recover from failure automatically and remains functional. Hadoop provides fault-tolerance at software level.

High Throughput

- Throughput is amount of data processed per unit of time. As you increase the number of nodes the throughput increases linearly (which is not true in RDBMS).



Flexible Software

- Dynamically add/remove nodes on the fly without disturbing/shutting down the cluster.

Cheap commodity hardware

- Need not use expensive, proprietary offerings from a single vendor.
- Commodity doesn’t mean our laptop/desktop machines which are cheap and higher failure rate.

Rack aware/Topology aware

- Hadoop is fed network topology (arrangement of nodes in a cluster) through configuration files.
- Hadoop can decide where to place data to minimize network flow and improve fault tolerance.

Batch processing centric

- Load data before you launch job.
- Job reads dataset from the beginning till the end, because there is no random read.
- need not wait for the result after job submission, because it is not interactive.

Shared nothing cluster computing

- One computer in a cluster cannot access the resources such as memory, storage… of another computer.

Effective to handle Volume and Variety**.**

Offered as a Cloud Service (HDInsight from azure cloud, EMR from amazon…)

free of cost as it is open source. Many companies have their own flavor of Hadoop by modifying source code.

Hadoop is not for all types of work

- not for transaction process (RDBMS is good for transactions). Because, index is given only for data blocks not for the content available in it.
- not for low latency process (because to run a job, it has to process entire big data).
- not good for large number of small files (because size of meta-data increases).
- not fair for compute intensive calculation that involves blocking, synchronization and communication among processes.
- not fair for interactive, iterative and stream processes.
- no support for multiple writers (file writing is continued at the end of file (appending), no arbitrary/random read and write).
- not good when work can’t be parallelized.

Hadoop is optimized to handle

- massive amount of data in parallel.
- structured/unstructured/semi structured data.
- inexpensive commodity hardware.

Hadoop programmer need not worry about

- managing cluster of nodes, file IO, networking…
- break file into blocks and where to distribute blocks.
- divide job into tasks and place computation near data.
- parallelization and load balancing…
- partial failure of a job: task/data/node failure (fault tolerance).

Hadoop takes care of complexities of distributed computing and lets programmers to concentrate writing algorithms to process data.

Companies using Hadoop

- **Google:** Index construction for google search, article clustering for google news…
- **Yahoo!:** “Web map” powering Yahoo! search, spam detection for Yahoo! Mail…
- **Facebook:** Ad optimization, spam detection…
- **Universities and research labs:** Astronomical image analysis (Washington), Bioinformatics (Maryland), Analyzing wikipedia conflicts (PARC), Natural language processing (CMU), Particle physics (Nebraska), Ocean climate simulation (Washington)…
- Head to https://wiki.apache.org/hadoop/PoweredBy for more information

Basic Terminologies

Technique represents how to perform a specific task. A variety of techniques may be used to solve a same problem.

Tools are programming languages used to implement techniques.

Technology is the scientific and engineering knowledge employed for building applications into usage and easing the task.

Framework/System/Platform consists stack/set of functions/components/tools working together and interact each other for a common goal.

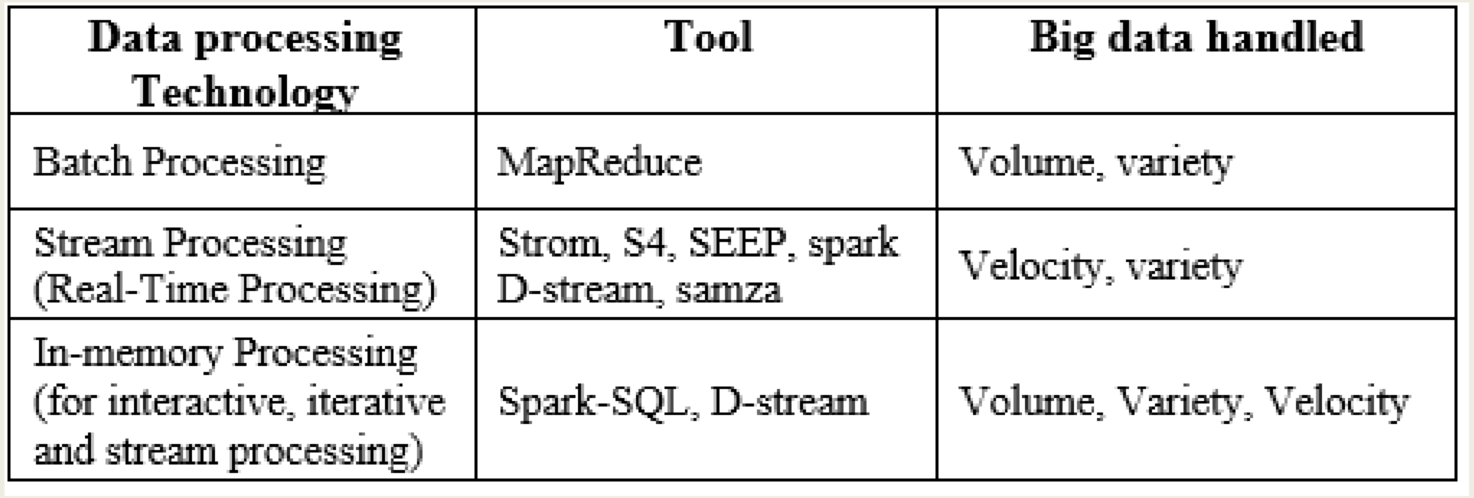
- Each function is a layer/black box providing abstraction (each layer knows only its input not the functionality of other layers) and may communicate with layer above and below.
- Failure of one layer doesn’t affect the other layer in the system. Ex: OSI, DBMS…

Now, we will see some examples for all these terms with respect to big data

- Big Data Techniques: any algorithms such as data mining for extracting unknown hidden pattern and machine learning for deriving actionable knowledge from big data.
- Big Data Tools: HDFS, Map-Reduce, Pig, Hive…
- Big Data Technologies: batch processing, stream processing, in-memory computing, NoSQL…
- Big Data Framework: Hadoop, Spark, Stratosphere, Storm…

Data Processing Technology

Data is processed in different fashion according to our requirement.



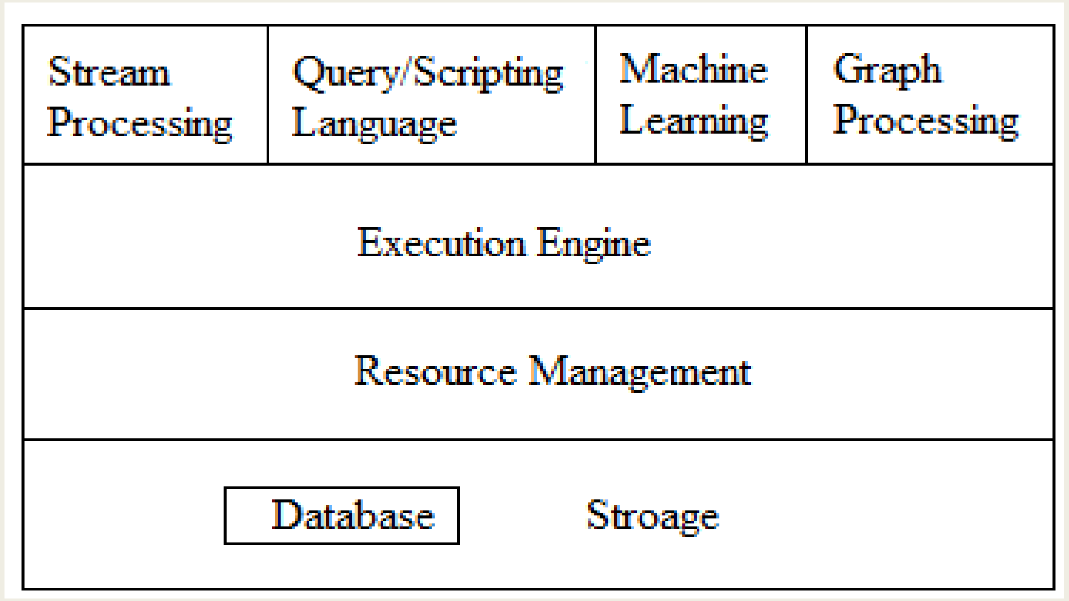
Data processing Technology	Tool	Big data handled
Batch Processing	MapReduce	Volume, variety
Stream Processing (Real-Time Processing)	Strom, S4, SEEP, spark D-stream, samza	Velocity, variety
In-memory Processing (for interactive, iterative and stream processing)	Spark-SQL, D-stream	Volume, Variety, Velocity

Big Data Framework

- provides set/stack of tools to process data in different fashion for decision making resulting in greater operational efficiencies, cost reductions, and reduced risks in businesses.

Big data analytics stack

- Every big data stack has tool to store data and process it in different fashion.
- The major components are storage, resource management and execution engine, upon which all other tools are installed to build applications.



Big data analytics stack

Big Data Analytics Stack Components

Storage

- Traditional file systems are not well-designed for large-scale data processing.
- Efficiency has a higher priority than other features.
- Massive size of data tends to store across multiple machines in a distributed way.
- Ex: HDFS, Amazon S3…

Database

- RDBMS was not designed to be distributed (but comes with huge cost due to synchronization and co-ordination).
- NoSQL databases relax one or more of the ACID properties (BASE) to achieve distributed, and scalable storage.
- Different data models: key/value, column-family, document, graph.
- Ex: Dynamo, Voldemort, Riak, Neo4J, BigTable, Hbase, Cassandra, MongoDB…

Resource management

- every framework is set up on dedicated cluster, because different frameworks require different computing resources.
- Deploying frameworks on dedicated cluster doesn’t utilize entire cluster resources all the time.
- Large organizations need the ability to share data and resources of cluster among multiple frameworks with clear isolation.
- Ex: Mesos, YARN, Quincy…

Execution engine

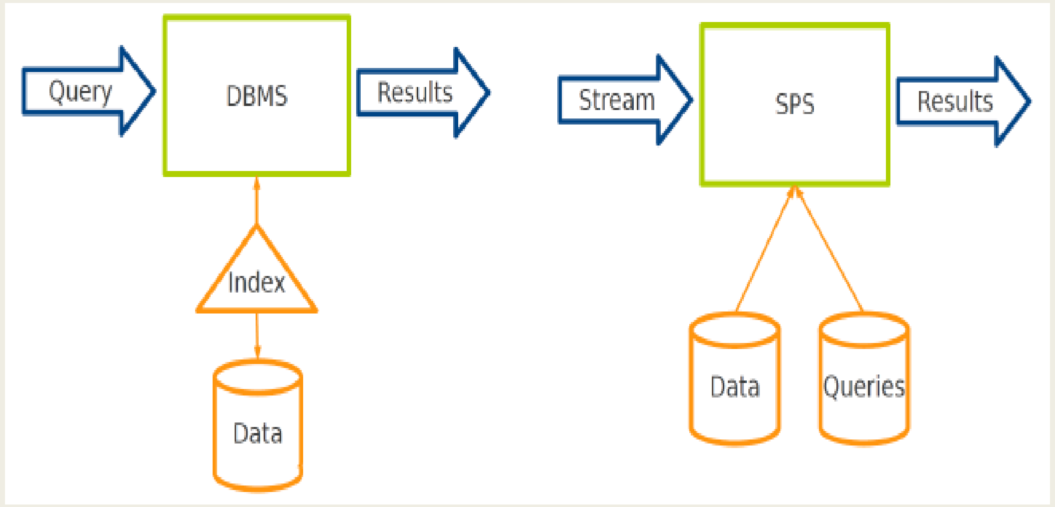
- should be distributed, scalable and fault tolerant data parallel processing on clusters of unreliable, commodity machines.
- Ex: MapReduce, Spark, Stratosphere, Dryad, Hyracks…

Query/Scripting language

- Low-level programming of execution engines such as MapReduce is not easy for non-java developers.
- Need high-level language to improve the query capabilities of execution engines.
- Ex: Pig, Hive, Shark, Meteor, DryadLINQ, SCOPE…
- 100 lines of MapReduce program can be written in 10 lines using Pig.
- A non-java programmer can directly use pig/hive without knowing MapReduce functionalities.
- These high-level pig/hive programs are translated into low-level MapReduce API of the execution engines at run time.

Stream processing system (SPS)

- processing incoming data before persistently stored and providing users with fresh, low latency results.
- Since data is processed in the memory itself there is no need of indexing mechanism.
- Ex: Storm, S4, SEEP, D-Stream, Naiad, Samza…

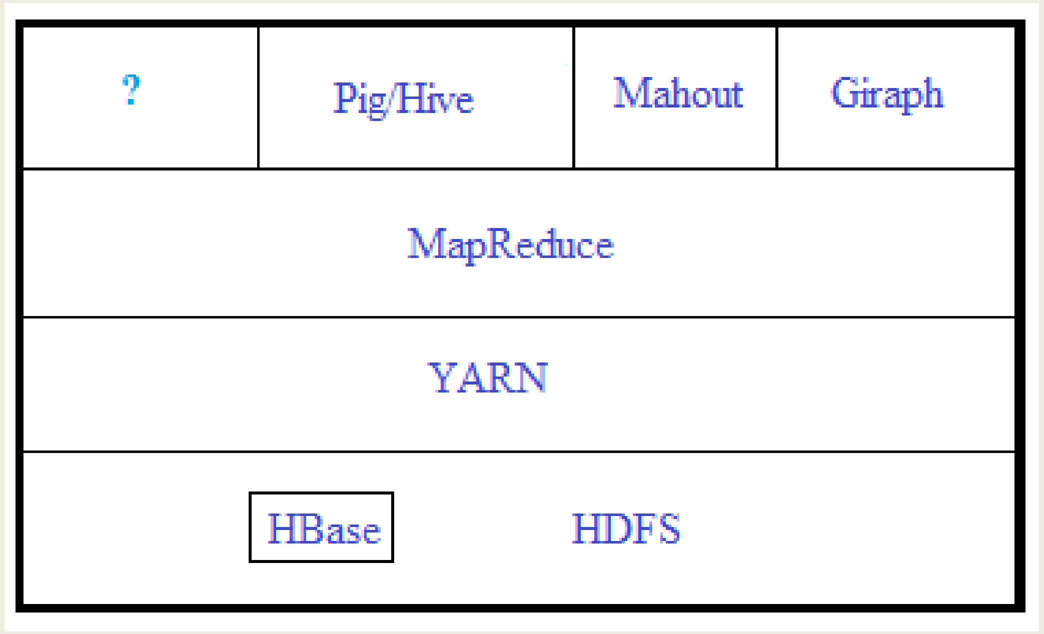


Graph processing

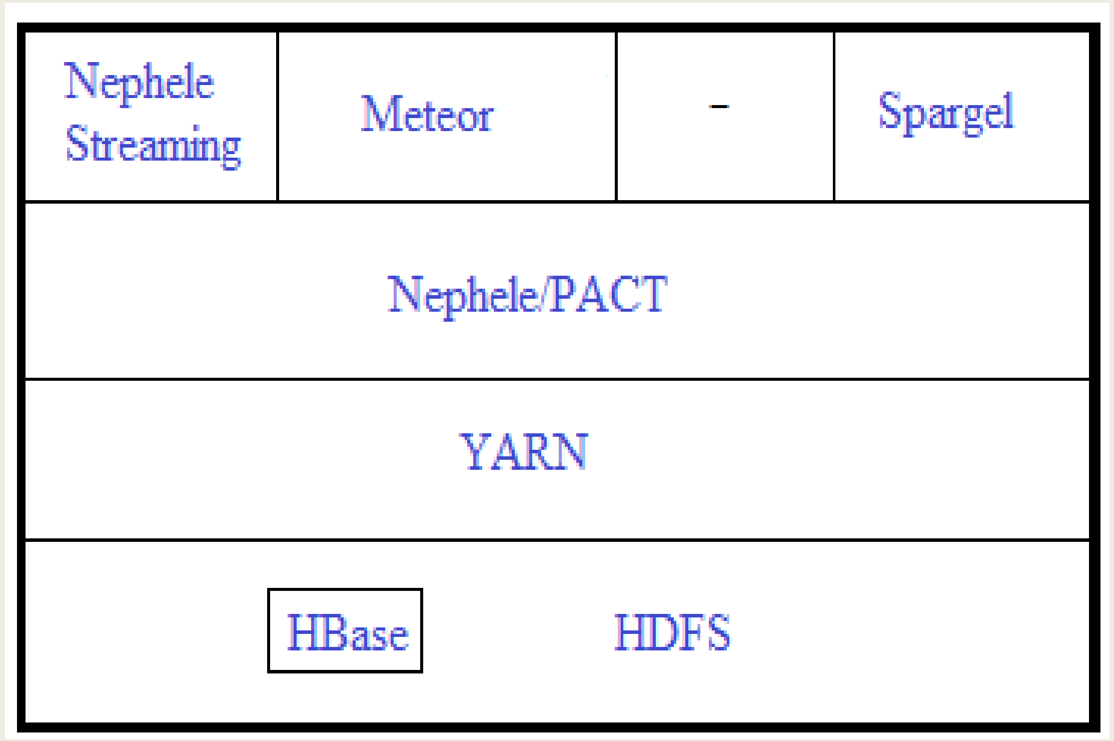
- Many problems are expressed as graphs and solved using graph algorithms.
- involves sparse computational dependencies, and multiple iterations to converge.
- Data-parallel framework such as MapReduce is not ideal for these problems: slow
- Graph processing frameworks given below are optimized for graph-based problems.
- Ex: Pregel, Giraph, GraphX, GraphLab, PowerGraph, GraphChi…

Machine learning

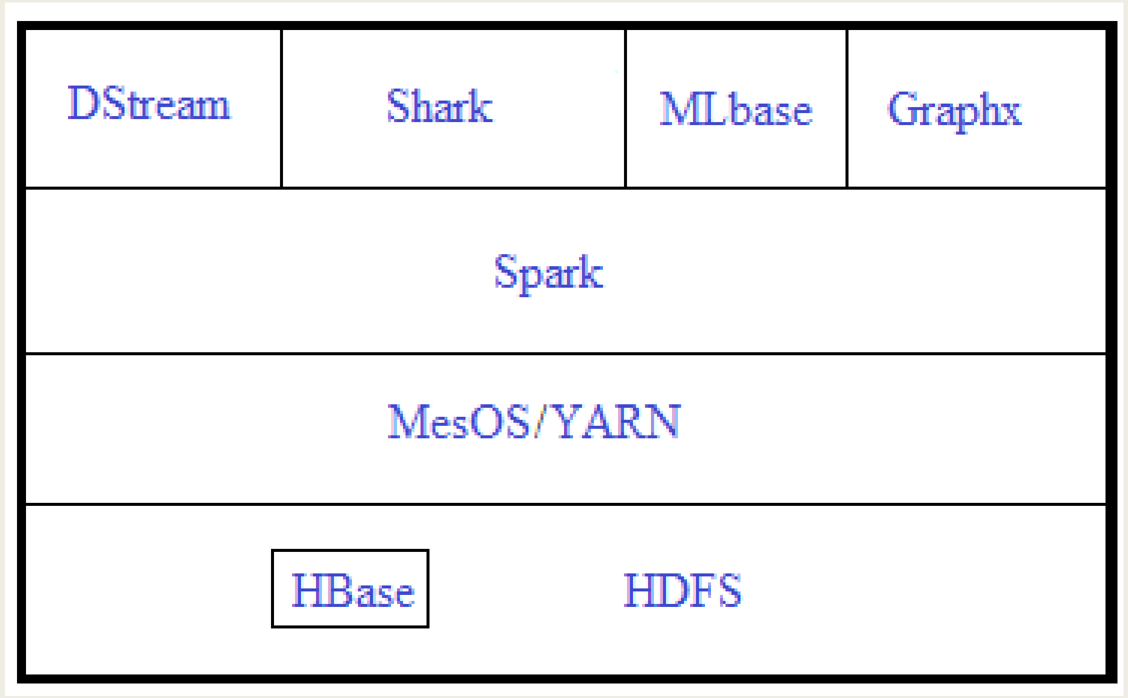
- Implementing and consuming machine learning techniques at scale are difficult tasks for developers and end users.
- There exist libraries that provide scalable machine learning and data mining algorithms.
- Ex: Mahout, MLBase, SystemML, Ricardo, Presto…



Hadoop Big Data analytics stack



Stratosphere Big Data analytics stack



Spark Big Data analytics stack

How to choose a right big data framework?

Decision depends on

Data size:

- If data fits into single system memory, then cluster is not required. GPU or multicore CPU can be used.
- If it doesn't fit in memory, then go for Hadoop, Spark…

Throughput: If batch processing is needed, then prefer any one of the scale out architectures such as MapReduce, Spark…

Data speed: If real-time response is required, then choose scale up architecture.

Machine learning: spark, stratosphere…

- To train model, use horizontal platform such as spark, MapReduce.
- To use model for real time response, use vertical platform.

Performance related metrics

Program gets an input and produces output. It is also called as job/task/…

- Performance – reducing latency of a job, measured with respect to one program.
- Throughput – amount of data processed or number of tasks processed per unit of time.
- Response time – time at which a job starts after submission.
- Execution time – amount of time a task spends actively with CPU (running time).
- Latency – time taken for a job from submission till completion. It includes execution time, intermediate delay. It is a terminology mainly used in networking.
- Turnaround time – used in place of latency in computer science.
- Makespan – time difference between the start and finish of a sequence of jobs. It is denoted as throughput in manufacturing.

Hadoop focuses on throughput rather than latency of a job.

Common pitfalls

data mining and big data same? No.

big data refers data that is massive, heterogeneous, and high-speed data while data mining refers to process of finding hidden pattern from those data.

Can we apply data mining on big data? Of course.

But, traditional data mining algorithms run in single machine. Therefore, we need to construct distributed and scalable algorithm that runs on many computers using tools like MapReduce, Spark, Storm…