THE UNIVERSITY OF MELBOURNE

Introduction to Big Data

Dr. Aaron Harwood
Department of Computing and Information Systems
University of Melbourne

Agenda

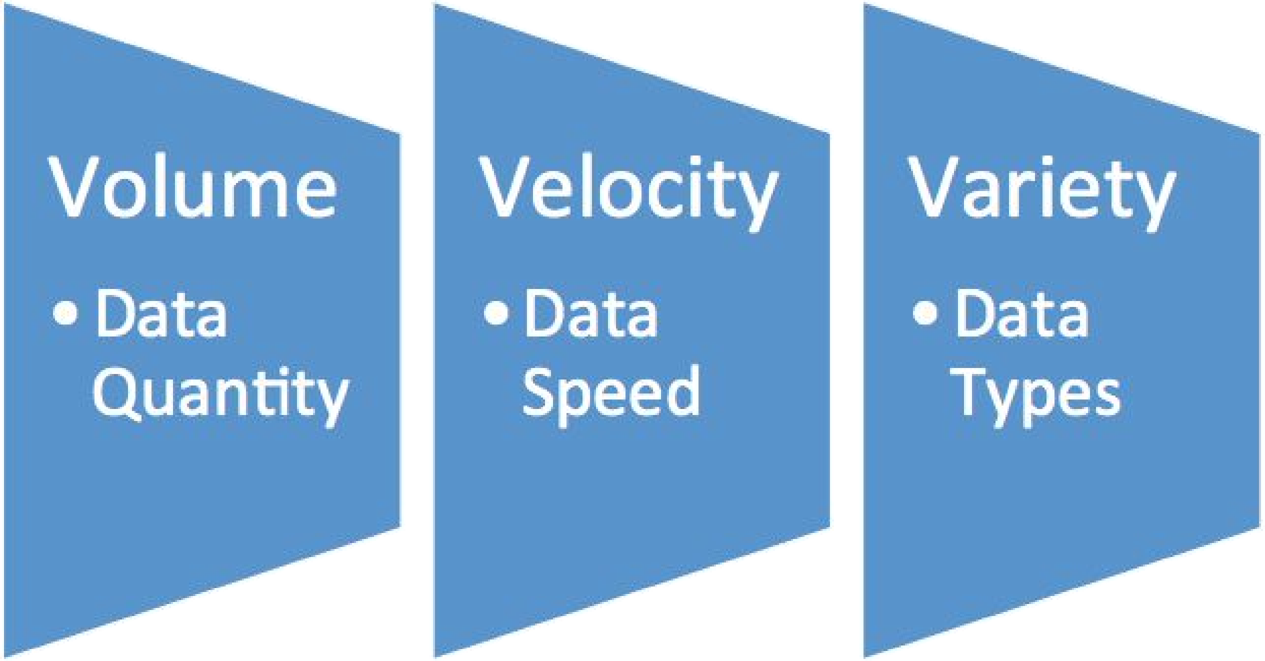
- What is Big Data?
- Big Data Characteristics
- Storing, Selecting and Processing of Big Data
- Why Big Data
- Application of Big Data analytics
- Challenges in Big Data Storage and Analysis
- Why Hadoop?
- Hadoop Origin and History
- Comparison between Hadoop and RDBMS
- Hadoop Architecture
- Hadoop Ecosystem overview

Big Data

- Think at Scale

  Data is in TB even in PB
  - Facebook has 400 terabytes of stored data and ingest 20 terabytes of new data per day. Hosts approx. 10 billion photos, 5PB(2011) and is growing 4TB per day
  - NYSE generates 1TB data/day
  - The Internet Archive stores around 2PB of data and is growing at a rate of 20PB per month
- Flood of data is coming from many resources
  - Social network profile, activity, logging and tracking
  - Public web information
  - Data warehouse appliances
  - Internet Archive store etc.

Big Data Characteristics - V3s



Volume

- Data Quantity

Velocity

- Data Speed

Variety

- Data Types

Big Data Characteristics - Volume

- A typical PC might have had 10 gigabytes of storage in 2000.
- Today, Facebook ingests 500 terabytes of new data every day.
- Boeing 737 will generate 240 terabytes of flight data during a single flight across the US.
- The smart phones, the data they create and consume; sensors embedded into everyday objects will soon result in billions of new, constantly-updated data feeds containing environmental, location, and other information, including video.

Big Data Characteristics - Velocity

- Clickstreams and ad impressions capture user behavior at millions of events per second
- High-frequency stock trading algorithms reflect market changes within microseconds
- Machine to machine processes exchange data between billions of devices
- Infrastructure and sensors generate massive log data in real-time
- Online gaming systems support millions of concurrent users, each producing multiple inputs per second.

Big Data Characteristics - Variety

- Big Data isn't just numbers, dates, and strings. Big Data is also geospatial data, 3D data, audio and video, and unstructured text, including log files and social media.
- Traditional database systems were designed to address smaller volumes of structured data, fewer updates or a predictable, consistent data structure.
- Big Data analysis includes different types of data

Storing Big Data

- Analyzing your data characteristics
  - Selecting data sources for analysis
  - Eliminating redundant data
  - Establishing the role of NoSQL
- Overview of Big Data stores
  - Data models: key value, graph, document, column-family
  - Hadoop Distributed File System
  - HBase
  - Hive

Selecting Big Data stores

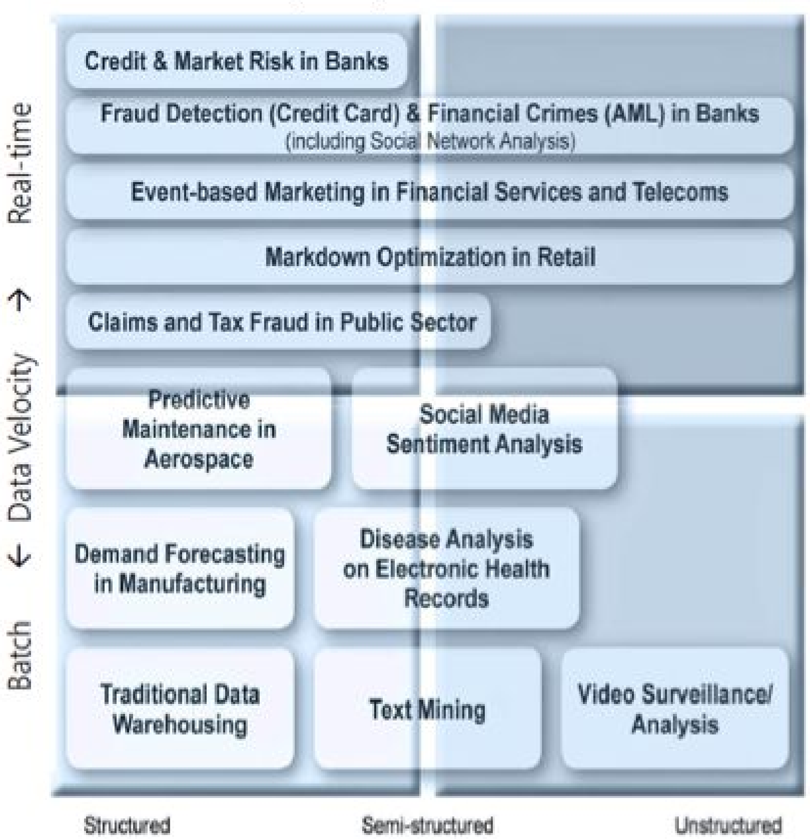
- Choosing the correct data stores based on your data characteristics
- Moving code to data
- Implementing polyglot data store solutions
- Aligning business goals to the appropriate data store

Processing Big Data

- Integrating disparate data stores
  - Mapping data to the programming framework
  - Connecting and extracting data from storage
  - Transforming data for processing
  - Subdividing data in preparation for Hadoop MapReduce
- Employing Hadoop MapReduce
  - Creating the components of Hadoop MapReduce jobs
  - Distributing data processing across server farms
  - Executing Hadoop MapReduce jobs
  - Monitoring the progress of job flows

The structure of Big Data

- Structured
  - Most traditional data sources
- Semi-structured
  - Many sources of big data
- Unstructured
  - Video data, audio data



Credit & Market Risk in Banks

Fraud Detection (Credit Card) & Financial Crimes (AML) in Banks (including Social Network Analysis)

Event-based Marketing in Financial Services and Telecoms

Markdown Optimization in Retail

Claims and Tax Fraud in Public Sector

Predictive Maintenance in Aerospace

Social Media Sentiment Analysis

Demand Forecasting in Manufacturing

Disease Analysis on Electronic Health Records

Traditional Data Warehousing

Text Mining

Video Surveillance/ Analysis

Batch ← Data Velocity → Real-time

Structured

Semi-structured

Unstructured

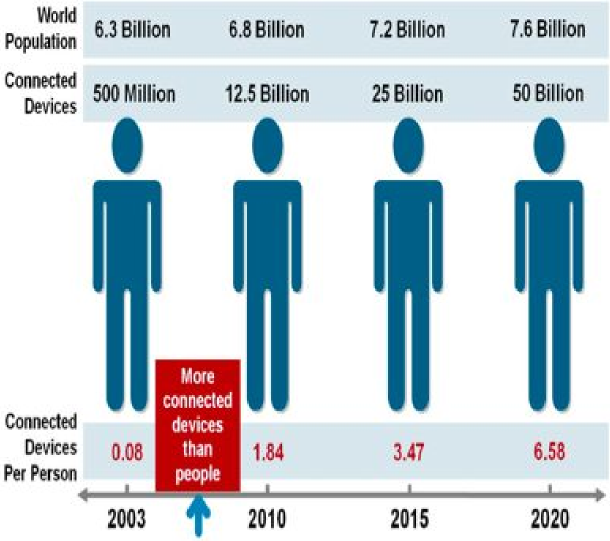
Why Big Data

- **Growth of Big Data is needed**
  - Increase of storage capacities
  - Increase of processing power
  - Availability of data (different data types)
  - Every day we create 2.5 quintillion bytes of data; 90% of the data in the world today has been created in the last two years alone

Why Big Data

- FB generates 10TB daily
- Twitter generates 7TB of data Daily
- IBM claims 90% of today's stored data was generated in just the last two years.

Figure 1. The Internet of Things Was "Born" Between 2008 and 2009



Source: Cisco IBSG, April 2011

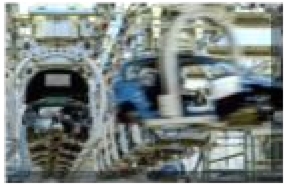
Application of Big Data analytics

Smarter Healthcare

Homeland Security

Traffic Control

Manufacturing



Multi-channel Sales

Telecom

Trading Analytics

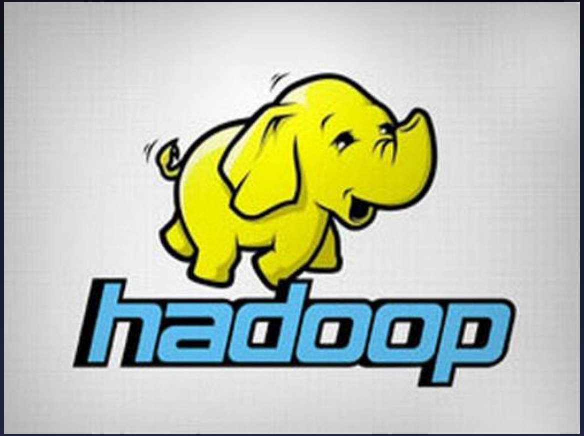
Search Quality

Challenges in Big Data Storage and Analysis

- Slow to process, cannot scale
  - Disk seek for every access
  - Buffered reads, locality → still seeking every disk page
  - It is not Storage Capacity but access speeds which is the bottleneck
  - Challenges to both store and analyze datasets
  - Scaling is expensive
- Hard Drive capacity to process
  - IDE drive - 75 MB/sec, 10ms seek
  - SATA drive - 300MB/s, 8.5ms seek
  - SSD - 800MB/s, 2ms “seek”
  - Apart from this, analyze, compute, aggregation, processing delay etc.
- Unreliable machines: Risk
  - 1 Machine 1 time in 3 years mean time between failures
  - 1000 Machines 1 day mean time between failures

Challenges in Big Data Storage and Analysis continues...

- Reliability
  - Partial failure, graceful decline rather than full halt
  - Data recoverability, if a node fails, another picks up its workload
  - Node recoverability, a fixed node can rejoin the group without a full group restart
  - Scalability, adding resources adds load capacity
- Backup
- Not affordable, expensive(faster, more reliability more cost)
- Easy to use and Secure
- Process data in parallel

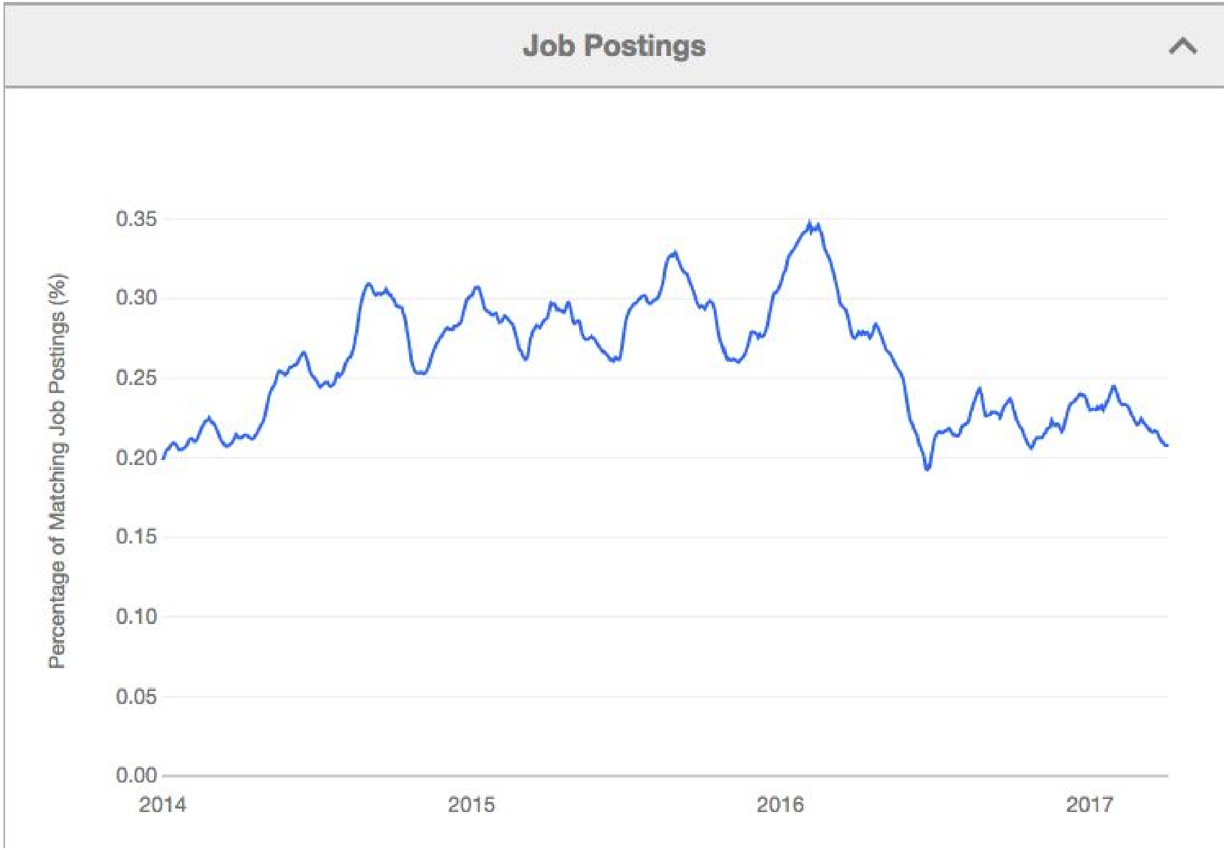


Why Hadoop?

Why Hadoop?

- Key features -
  - Flexible
  - Scalable
  - Building more efficient data economy
  - Robust Ecosystem
  - Hadoop is getting more “Real-Time”
  - Cost Effective
  - Upcoming Technologies using Hadoop
  - Hadoop is getting Cloudy!

Forecast growth of Hadoop Job Market



Source

Hadoop origins

Apache Hadoop is a framework that allows for the distributed processing of large data sets across clusters of commodity computers using a simple programming model. It is designed to scale up from single servers to thousands of machines, each providing computation and storage.

- Hadoop is an open-source implementation of Google MapReduce, GFS (distributed file system)
- Hadoop was created by Doug Cutting, the creator of Apache Lucene, the widely used text search library
- Hadoop fulfill need of common infrastructure
  - Efficient, reliable, easy to use
  - Open Source, Apache License

The Name ‘Hadoop’

- When naming software projects, Doug Cutting seems to have been inspired by his family.
- His son, as a toddler, used Nutch as the all-purpose word for meal and later named a yellow stuffed elephant Hadoop.
- Doug said he ‘was looking for a name that wasn’t already a web domain and wasn’t trademarked, so I tried vairous words that were in my life but not used by anybody else. Kids are pretty good at marking up words’.

Hadoop Design Axioms

- Store and process large amounts of data (PetaBytes)
- Performance, storage, processing scale linearly
- Computation should move to data
- Simple code, modular and extensible
- Failure is normal, expected
- Manageable and Heal self
- Design run on commodity hardware - cost effective

For Storage and Distributed computing (MapReduce)

- Split up the data
- Process Data in parallel
- Sort and combine to get the answer
- Schedule, Process and Aggregate independently
- Failures are independent, Handle failures
- Handle fault tolerance

Hadoop History

- 2002- 2004 Doug Cutting and Mike Cafarella started working on Nutch
- 2003- 2004: Google publishes GFS and MapReduce paper
- 2004: Doug Cutting adds DFS and MapReduce support to Nutch
- Yahoo! Hires Cutting, build team to develop Hadoop
- 2007: NY time converts 4TB of archive over 100 EC2 cluster of Hadoop
- Web scale deployment at Y!, Facebook, twitter
- May 2009: Yahoo does fastest sort of a TB, 62 secs over 1460 nodes
- Yahoo sort a PB in 16.25hrs over 3658 nodes

Hadoop v/s RDBMS

An Elephant can't jump. But can carry heavy load!

- A fundamental tenet of relational database structure defined by a schema, what about large data sets are often unstructured or semi-structured, Hadoop is the best choice. Hadoop MR framework uses key/value pairs as its basic data unit, which is flexible enough to work with the less-structured data types
- **Scaling commercial relational databases is expensive and limited.**
- High-level declarative language like SQL, Block box Query engine. You query data by stating the result you want and let the database engine figure and drive it. You can build **complex statistical models from your data or analytical reporting** or reformat your image data. SQL is not well designed for such tasks. MapReduce tries to collocate the data with the compute node, so data access is fast since it is local.

Hadoop v/s RDBMS continues

- To run a bigger database you need to buy a bigger machine. The high-end machines are not cost effective for many applications. E.g., a machine with four times the power of a standard PC costs a lot more than putting four such PCs in a cluster. Hadoop is designed to be a **scale-out architecture** operating on a cluster of commodity hardware. Adding more resources means adding more machines to the Hadoop cluster.
  - Effective cost per user TB: $250/TB
  - Other solutions (RDBMS) cost in the range of $100 to $100K per user TB
- **Hardest aspect is gracefully handling partial failure** -- when you don't know if a remote process has failed or not-- and still making progress with the overall computation. MapReduce spares the programmer from having to think about failure, since the implementation detects failed map or reduce tasks and reschedules with suitable replacements. MapReduce is able to do this since it is a **shared-nothing architecture,** meaning that the tasks have no dependence on one other.

Is Hadoop alternative for RDBMS?

Hadoop is not replacing the traditional data systems used for building analytic applications - the RDBMS, EDW and MPP systems - but rather is a complement

- Interoperate with existing systems and tools, at the moment Apache Hadoop is not a substitute for a database
- No Relation, Key Value Pairs
- Big Data, unstructured (Text) & semi structured (Seq / Binary Files)
- Structured (Hbase = Google BigTable)
- Works fine together with RDBMS, Hadoop is being used to large quantities of data into something more manageable.

Hadoop Architecture

- **Hadoop designed and built on two independent frameworks** Hadoop = HDFS + MapReduce

  HDFS (storage and File system): HDFS is a reliable distributed file system that provides high-throughput access to data

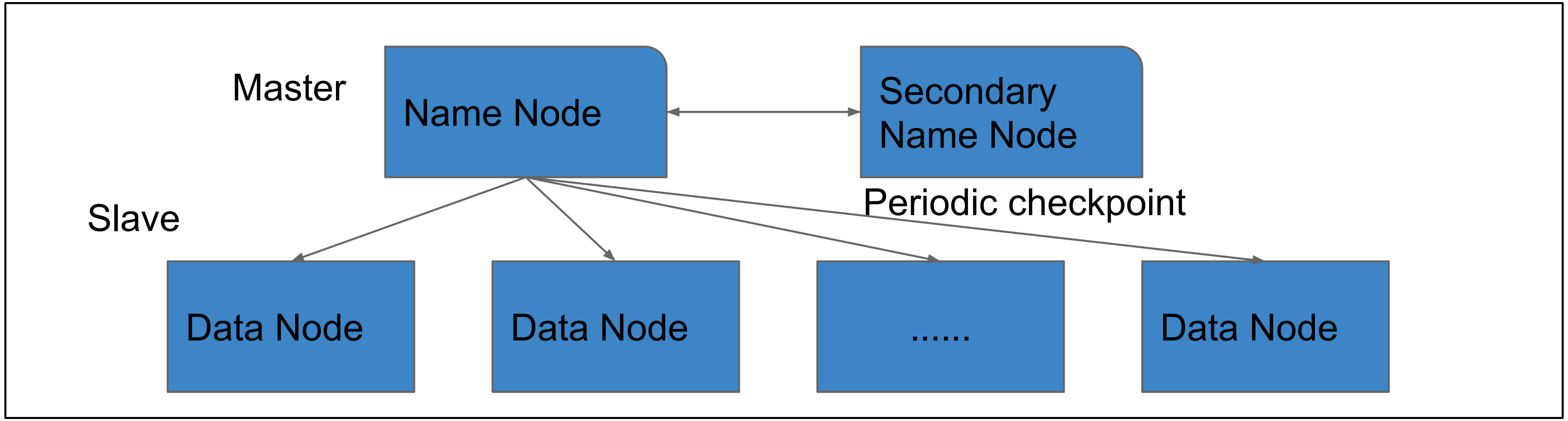
  MapReduce (processing): MapReduce is a framework for performing high performance distributed data processing using the divide and aggregate programming paradigm

- **Hadoop has a master/slave architecture for both storage and processing**

Hadoop Master and Slave Architecture

The components (daemons) of Hadoop Distributed File Systems (HDFS) are

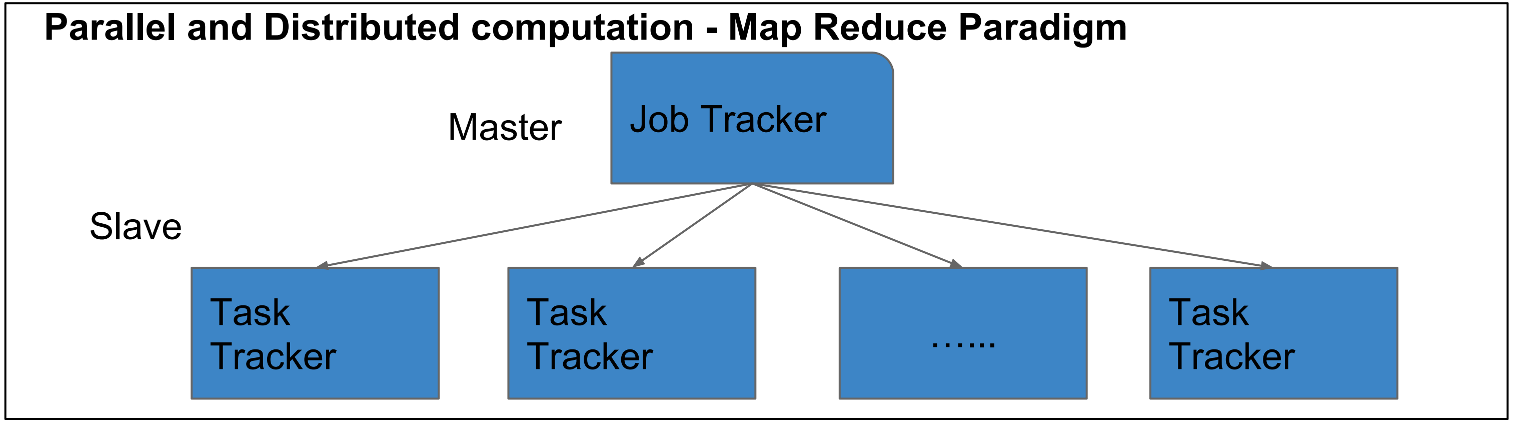
- **NameNode** is the master of the system. It **maintains the name system** (directories and files) and manages the blocks which are present on the DataNodes.
- **DataNodes** are the slaves which are deployed on each machine and provide the actual storage. They are responsible for serving read and write requests for the clients.
- **Secondary NameNode** is responsible for **performing periodic checkpoints**. So, in the event of NameNode failure, you can restart the NameNode using the checkpoint.



Hadoop Master and Slave Architecture continues...

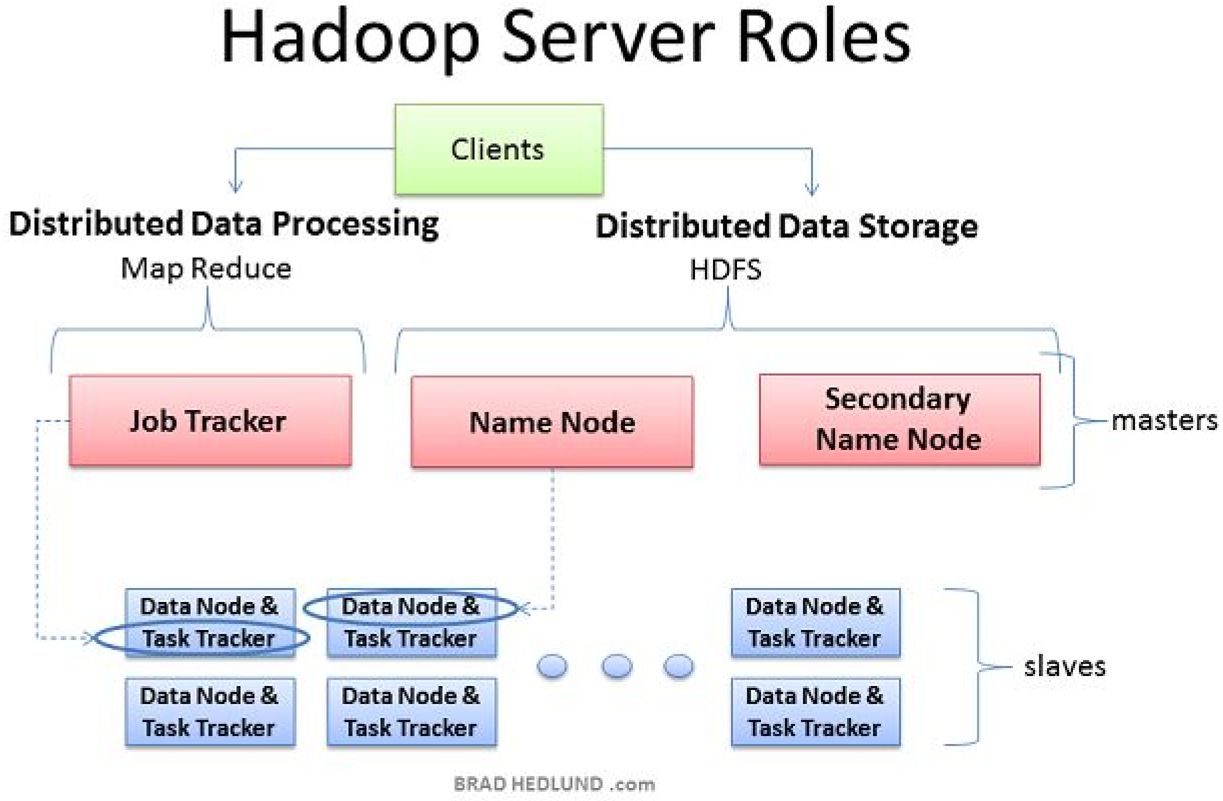
The components (daemons) of MapReduce are:

- **JobTracker** is the master of the system which manages the jobs and resources in the cluster (TaskTrackers). The JobTracker tries schedule each map as close to the actual data being processed i.e. on the TaskTracker which is running on the same DataNode as the underlying block.
- **TaskTrackers** are the slaves which are deployed on each machine. They are responsible for running the map and reduce tasks as instructed by the JobTracker



Parallel and Distributed computation - Map Reduce Paradigm

Core Hadoop Ecosystem



Hadoop Server Roles

Clients

Distributed Data Processing

Map Reduce

Distributed Data Storage

HDFS

Job Tracker

Name Node

Secondary Name Node

masters

Data Node & Task Tracker

Data Node & Task Tracker

Data Node & Task Tracker

Data Node & Task Tracker

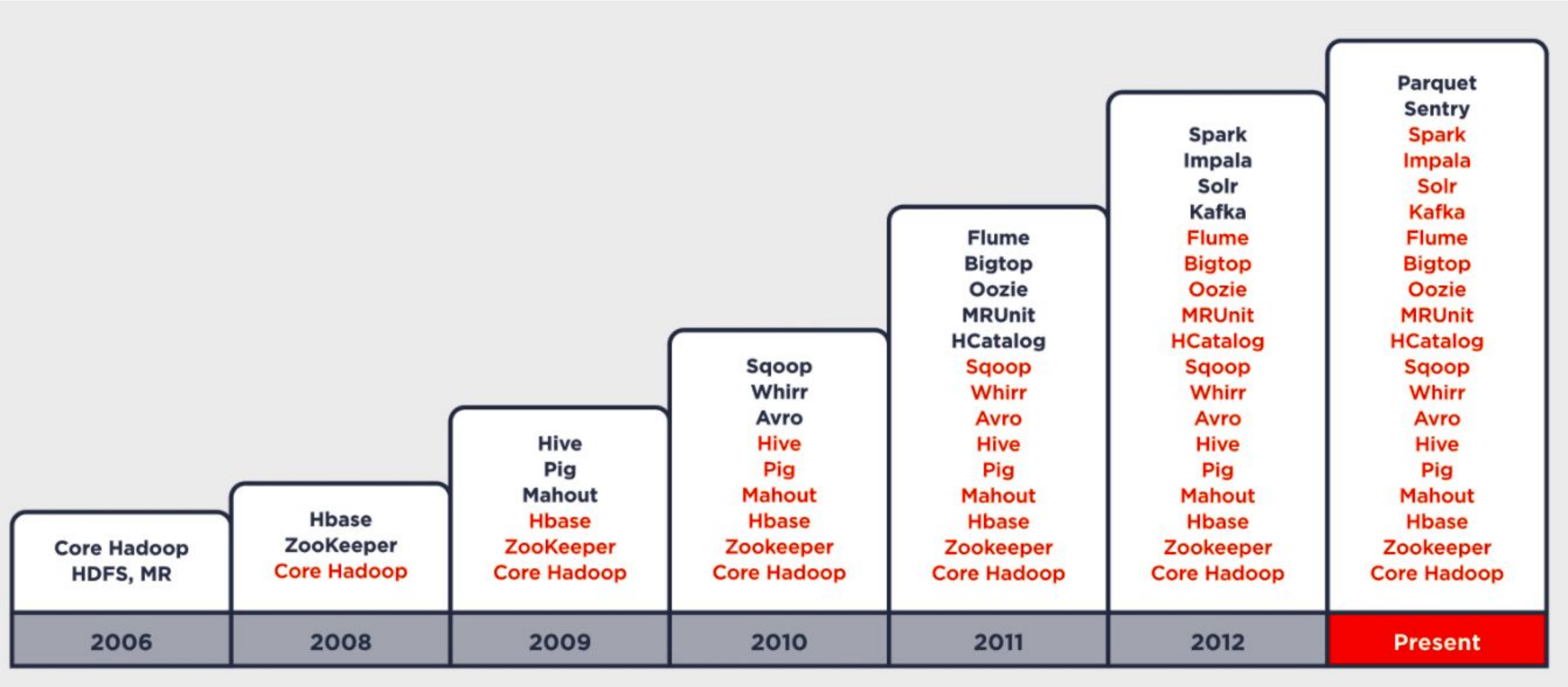
Data Node & Task Tracker

Data Node & Task Tracker

slaves

BRAD HEDLUND .com

Hadoop Ecosystem Development

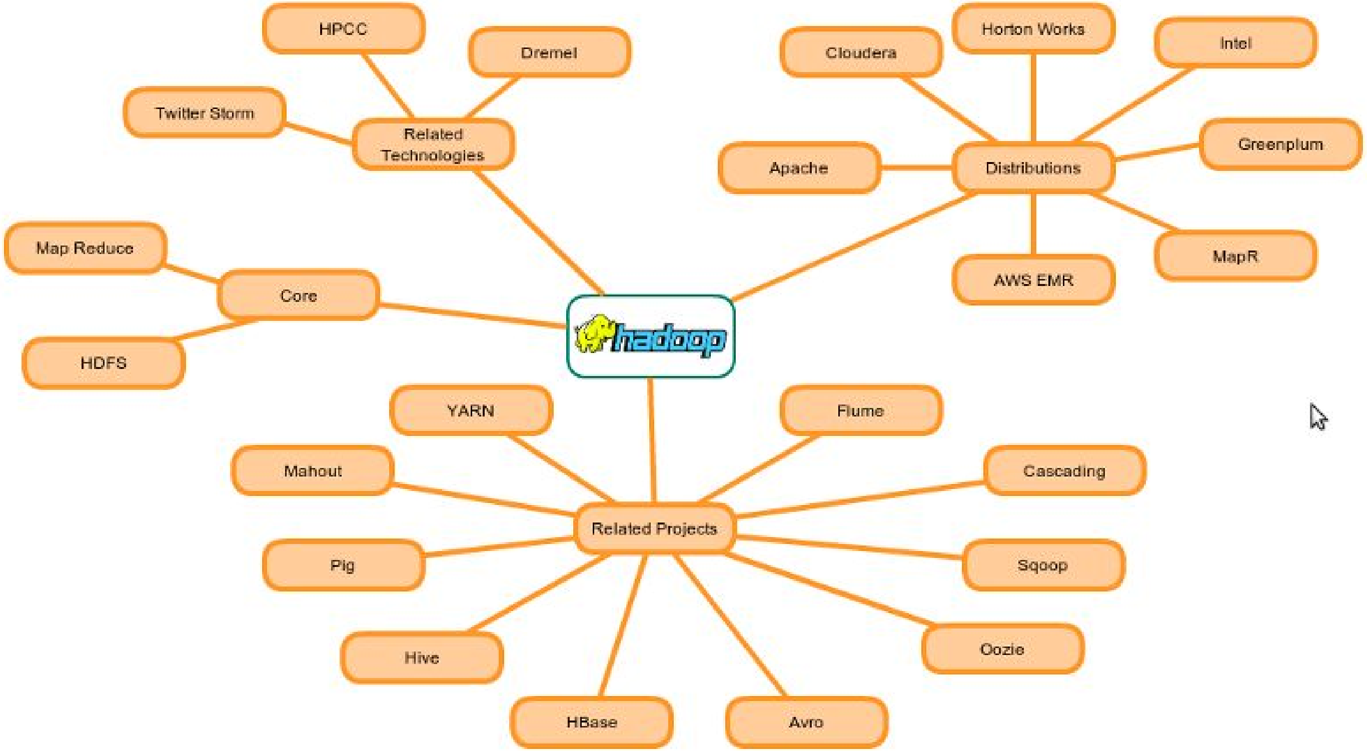


Core Hadoop HDFS, MR	Hbase ZooKeeper Core Hadoop	Hive Pig Mahout Hbase ZooKeeper Core Hadoop	Sqoop Whirr Avro Hive Pig Mahout Hbase Zookeeper Core Hadoop	Flume Bigtop Oozie MRUnit HCatalog Sqoop Whirr Avro Hive Pig Mahout Hbase Zookeeper Core Hadoop	Spark Impala Solr Kafka Flume Bigtop Oozie MRUnit HCatalog Sqoop Whirr Avro Hive Pig Mahout Hbase Zookeeper Core Hadoop	Parquet Sentry Spark Impala Solr Kafka Flume Bigtop Oozie MRUnit HCatalog Sqoop Whirr Avro Hive Pig Mahout Hbase Zookeeper Core Hadoop
2006	2008	2009	2010	2011	2012	Present

Source:

Hadoop Ecosystem Now…

The Hadoop ecosystem has grown over the last few years and there is a lot of jargon in terms of tools, frameworks. Hadoop has become the kernel.



Hadoop Environment

Hadoop can be configured in three modes

- **Standalone:** Hadoop all Daemons in run inside a single Java process. Use local file for storage. Standalone mode helpful for debug Hadoop applications.
- **Pseudo-distributed:** Each Hadoop daemon runs in different JVM, as a separate process, but all processes running on a single machine.
- **Fully-distributed:** Hadoop actual powers parallel processing, scalability and the independence of task execution, replication management, workflow management, fault-tolerance, and data consistency are lies in the fully distributed mode. The Hadoop fully distributed mode is highly effective centralized data structure allows multiple machines to contribute processing power and storage to the cluster.

Hadoop Features and summary

Distributed framework for processing and storing data generally on commodity hardware. Completely open source and Written in Java

- **Store anything**
  - Unstructured or semi structured data
- **Storage capacity**
  - Scale linearly, cost is not exponential
- **Data locality and process in your way**
  - Code moves to data
  - In MR you specify the actual steps in processing the data and drive the output. Stream access: Process data in any language
- **Failure and fault tolerance**
  - Detect failures and heals itself
  - Reliable, data replicated, failed task are rerun, no need to maintain backup of data
- **Cost effective**: Hadoop is designed to be a scale-out architecture operating on a cluster of commodity PC machines
- The Hadoop framework transparently for customization to provide applications reliability, adaption and data motion

Primarily used for batch processing, not real-time/transactional user applications.

What is Hadoop used for?

- Search
  - Yahoo, Amazon, Zvents
- Log processing
  - Facebook, Yahoo, ContextWeb, Joost, Last.fm
- Recommendation Systems
  - Facebook
- Data Warehouse
  - Facebook, AOL
- Video and Image Analysis
  - New York Times, Eyealike

…… Almost in every domain!!

Who uses Hadoop?

- Amazon/A9
- Facebook
- Google
- IBM
- Disney
- Last.fm
- New York Times
- PowerSet
- Veoh
- Yahoo!
- Twitter
- LinkedIn
- …

References

- Hadoop: The Definitive Guide, Third Edition by Tom White
- http://hadoop.apache.org/
- https://www.cloudera.com/
- https://www.slideshare.net/sudhakara_st/hadoop-intruduction
-

Thanks!

Any questions?

1

BACK UP