Big Data Management

What is Big Data?

- No standard definition! here is from Wikipedia:
  - Big data is a term for data sets that are so large or complex that traditional data processing applications are inadequate.
  - Challenges include analysis, capture, data curation, search, sharing, storage, transfer, visualization, querying, updating and information privacy.
  - Analysis of data sets can find new correlations to "spot business trends, prevent diseases, combat crime and so on."

Who is generating Big Data?

Social

User Tracking & Engagement

Homeland Security

eCommerce

Financial Services

Real Time Search

Big Data Characteristics: 3V

***Source**: Contents of above graphic created in partnership with Teradata, Inc.*

Volume (Scale)

- Data Volume
  - Growth 40% per year
  - From 8 zettabytes (2016) to 44zb (2020)
- Data volume is increasing exponentially

Exponential increase in collected/generated data

Processes 20 PB a day (2008)
Crawls 20B web pages a day (2012)
Search index is 100+ PB (5/2014)
Bigtable serves 2+ EB, 600M QPS (5/2014)

400B pages, 10+ PB (2/2014)

Hadoop: 365 PB, 330K nodes (6/2014)

150 PB on 50k+ servers running 15k apps (6/2011)

Hadoop: 10K nodes, 150K cores, 150 PB (4/2014)

300 PB data in Hive + 600 TB/day (4/2014)

S3: 2T objects, 1.1M request/second (4/2013)

LHC: ~15 PB a year

LSST: 6-10 PB a year (~2020)

640K ought to be enough for anybody.

SKA: 0.3 – 1.5 EB per year (~2020)

How much data?

Variety (Complexity)

- Different Types:
  - Relational Data (Tables/Transaction/Legacy Data)
  - Text Data (Web)
  - Semi-structured Data (XML)
  - Graph Data
    - Social Network, Semantic Web (RDF), …
  - Streaming Data
    - You can only scan the data once
  - A single application can be generating/collecting many types of data
- Different Sources:
  - Movie reviews from IMDB and Rotten Tomatoes
  - Product reviews from different provider websites

To extract knowledge➔ all these types of data need to linked together

A Single View to the Customer

A Global View of Linked Big Data

Diversified social network

Heterogeneous information network

Velocity (Speed)

- Data is begin generated fast and need to be processed fast
- Online Data Analytics
- Late decisions ➔ missing opportunities
- Examples
  - **E-Promotions:** Based on your current location, your purchase history, what you like ➔ send promotions right now for store next to you
  - **Healthcare monitoring:** sensors monitoring your activities and body ➔ any abnormal measurements require immediate reaction
  - **Disaster management and response**

Real-Time Analytics/Decision Requirement

Product Recommendations that are ***Relevant*** & ***Compelling***

Learning why Customers Switch to competitors and their offers; in time to Counter

Improving the Marketing Effectiveness of a Promotion while it is still in Play

Friend Invitations to join a Game or Activity that expands business

Preventing Fraud as it is ***Occurring*** & preventing more proactively

Extended Big Data Characteristics: 6V

- Volume: In a big data environment, the amounts of data collected and processed are much larger than those stored in typical relational databases.
- Variety: Big data consists of a rich variety of data types.
- Velocity: Big data arrives to the organization at high speeds and from multiple sources simultaneously.

- Veracity: Data quality issues are particularly challenging in a big data context.
- Visibility/Visualization: After big data being processed, we need a way of presenting the data in a manner that's readable and accessible.
- Value: Ultimately, big data is meaningless if it does not provide value toward some meaningful goal.

Veracity (Quality & Trust)

- *Data = quantity + quality*
- When we talk about big data, we typically mean its quantity:
  - What capacity of a system provides to cope with the sheer size of the data?
  - Is a query feasible on big data within our available resources?
  - How can we make our queries tractable on big data?
  - . . .
- Can we trust the answers to our queries?
  - Dirty data routinely lead to misleading financial reports, strategic business planning decision $\Rightarrow$ loss of revenue, credibility and customers, disastrous consequences
- *The study of data quality is as important as data quantity*

Data in real-life is often dirty

81 million National Insurance numbers but only 60 million eligible citizens

98000 deaths each year, caused by errors in medical data

500,000 dead people retain active Medicare cards

Visibility/Visualization

- Visible to the process of big data management
- Big Data – visibility = Black Hole?

A visualization of Divvy bike rides across Chicago

- Big data visualization tools:

BI/Visualization/ Analytics

pentaho COGNOS tableau sas DOMO looker ClearStory QlikView MicroStrategy Business Objects

Value

- Big data is meaningless if it does not provide value toward some meaningful goal

Big Data: 6V in Summary

Transforming Energy and Utilities through Big Data & Analytics. By Anders Quitzau@IBM

Other V's

- Variability
  - Variability refers to data whose meaning is constantly changing. This is particularly the case when gathering data relies on language processing.
- Viscosity
  - This term is sometimes used to describe the latency or lag time in the data relative to the event being described. We found that this is just as easily understood as an element of Velocity.
- Virality
  - Defined by some users as the rate at which the data spreads; how often it is picked up and repeated by other users or events.
- Volatility
  - Big data volatility refers to how long is data valid and how long should it be stored. You need to determine at what point is data no longer relevant to the current analysis.
- More V's in the future …

Big Data Tag Cloud

BIG
DATA
INFORMATION
STORAGE
SETS
SEARCH
PETABYTES
ANALYSIS
RESEARCH
INTERNET
MANAGEMENT
TECHNOLOGIES
GOVERNMENT
PROCESS
VOLUME
LARGE
SOFTWARE
SAN
NAS
MILLION
SCIENCE
FUTURE
DEVELOPMENT
TECHNOLOGY
BILLION
NEW
EXABYTES
SIZE
DATABASES
PROCESSING
AMOUNT
APPLICATIONS
LEARNING
WORLD
BASED
EVERY
SOCIAL
YEARS
DRIVE
REPLICATION
SOURCES
LIMITS
CHALLENGES
COMPLEX
ANNOUNCED
PAST
INCLUDE
NETWORKS
HYPOTHESIS
END
MAPREDUCE
CRITIQUES
PARALLEL
STATE
CRITIQUE
PEOPLE
TRAFFIC
CURRENTLY
TERABYTES
MARKET
VARIETY
RELATED
TIME
SENSOR
WORLD
NOW
USED
INITIATIVE
ECONOMIC
PROJECT
DISTRIBUTED
RELEVANT
PEER
RECORDS
COST
INSIGHT
TOOLS
CENTER
PARADIGM
FUNDING
ARCHITECTURE
PROCESSES
APPROACHES
PRIVACY
INTERNATIONAL
IMPLEMENTATION
BECOMES
ONE
OUT
SECOND
NATIONAL
FLOW
INTELLIGENCE
APPROACH
MANY
USE
NEARLY
MOST
NEW
FRAMEWORK
BUSINESS
EFFECTIVE
VISUALIZATION
STATISTICS
DEFINITION
BETWEEN
MASSIVE
USING
FACTOR
MANAGE
ONLINE
DAY
WORLDWIDE
SYSTEMS
MAKING
INTEGRATION
SIMULATIONS
ANALYTICS
HIGHER
COMPANIES
ALGORITHMS
HANDLE
HAND
CAPACITY
ONE
PRIVATE
LESS
WIDE
EVERY
VARIOUS
TIMES
SHARED
NEEDED
USES
COMPARED
SUCCESSFUL
ABILITIES
YEAR
WORK
INFRASTRUCTURE
APPLICATIONS
AMOUNT
RATE
STRUCTURE
REAL
QUERIES
HIGH
USERS
DEPARTMENT
SET
WELL
DETERMINE
TERMS
RESULTS
DRIVE
SOURCES
DAY
WORLDWIDE
MANAGE
ONLINE
MASSIVE
USING
FACTOR
SYSTEMS
STATISTICS
DEFINITION
BETWEEN
EFFECTIVE
VISUALIZATION
WELL
SET
DEPARTMENT
QUERIES
HIGH
USERS
REAL
AMOUNT
RATE
STRUCTURE
WORK
APPLICATIONS
INFRASTRUCTURE
ANALYTICS
SIMULATIONS
HIGHER
COMPANIES
ALGORITHMS
EXABYTES
SIZE
PROCESSING
DATABASES
BUSINESS
FRAMEWORK
MANAGEMENT
VOLUME
PROCESS
GOVERNMENT
TECHNOLOGIES
SAN
NAS
SOFTWARE
LARGE
STORAGE
MILLION
SETS
SCIENCE
FUTURE
DEVELOPMENT
TECHNOLOGY
BILLION
NEW
CRITIQUES
PARALLEL
STATE
CRITIQUE
PEOPLE
TRAFFIC
CURRENTLY
TERABYTES
MARKET
VARIETY
RELATED
TIME
SENSOR
WORLD
NOW
USED
INITIATIVE
ECONOMIC
PROJECT
DISTRIBUTED
RELEVANT
PEER
RECORDS
COST
INSIGHT
TOOLS
CENTER
PARADIGM
FUNDING
ARCHITECTURE
PROCESSES
APPROACHES
PRIVACY
INTERNATIONAL
IMPLEMENTATION
BECOMES
ONE
OUT
SECOND
NATIONAL
FLOW
INTELLIGENCE
APPROACH
MANY
USE
NEARLY
MOST
MAPREDUCE
INCLUDE
NETWORKS
HYPOTHESIS
END
SCIENCE
CHALLENGES
COMPLEX
ANNOUNCED
PAST
LIMITS

Why Study Big Data Technologies?

- The hottest topic in both research and industry
- Highly demanded in real world
- A promising future career
  - Research and development of big data systems:

    distributed systems (eg, Hadoop), visualization tools, data warehouse, OLAP, data integration, data quality control, …
  - Big data applications:

    social marketing, healthcare, …
  - Data analysis: to get values out of big data

    discovering and applying patterns, predicative analysis, business intelligence, privacy and security, …

Big Data Open Source Tools

Philosophy to Scale for Big Data Processing

Divide Work

Combine Results

Distributed processing is non-trivial

- How to assign tasks to different workers in an efficient way?
- What happens if tasks fail?
- How do workers exchange results?
- How to synchronize distributed tasks allocated to different workers?

Big data storage is challenging

- Data Volumes are massive
- Reliability of Storing PBs of data is challenging
- All kinds of failures: Disk/Hardware/Network Failures
- Probability of failures simply increase with the number of machines …

What is Hadoop

- Open-source data storage and processing platform
- Before the advent of Hadoop, storage and processing of big data was a big challenge
- Massively scalable, automatically parallelizable
  - Based on work from Google
    - Google: GFS + MapReduce + BigTable (Not open)
    - Hadoop: HDFS + Hadoop MapReduce + HBase（opensource)
- Named by Doug Cutting in 2006 (worked at Yahoo! at that time), after his son's toy elephant.

Hadoop offers

- Redundant, Fault-tolerant data storage
- Parallel computation framework
- Job coordination

Programmers

No longer need to worry about

Q: Where file is located?

Q: How to handle failures & data lost?

Q: How to divide computation?

Q: How to program for scaling?

Why Use Hadoop?

- Cheaper
  - Scales to Petabytes or more easily
- Faster
  - Parallel data processing
- Better
  - Suited for particular types of big data problems

Companies Using Hadoop

facebook

The New York Times

ebay

hp

IBM

JPMorganChase

eHarmony

twitter

intel

NETFLIX

rackspace HOSTING

amazon.com

VISA

NING

SAMSUNG

YAHOO!

Forecast growth of Hadoop Job Market

Hadoop is a set of Apache Frameworks and more...

- Data storage (HDFS)
  - Runs on commodity hardware (usually Linux)
  - Horizontally scalable
- Processing (MapReduce)
  - Parallelized (scalable) processing
  - Fault Tolerant
- Other Tools / Frameworks
  - Data Access
    - HBase, Hive, Pig, Mahout
  - Tools
    - Hue, Sqoop
  - Monitoring
    - Greenplum, Cloudera

What are the core parts of a Hadoop distribution?

HDFS Storage

Redundant (3 copies)

For large files – large blocks

64 or 128 MB / block

Can scale to 1000s of nodes

MapReduce API

Batch (Job) processing

Distributed and Localized to clusters (Map)

Auto-Parallelizable for huge amounts of data

Fault-tolerant (auto retries)

Adds high availability and more

Other Libraries

Pig

Hive

HBase

Others

Hadoop 2.0

- Single Use System
  - Batch apps
- Multi-Purpose Platform
  - Batch, Interactive, Online, Streaming

Hadoop YARN *(Yet Another Resource Negotiator)*: a resource-management platform responsible for managing computing resources in clusters and using them for scheduling of users' applications

Hadoop Ecosystem

A combination of technologies which have proficient advantage in solving business problems.

http://www.edupristine.com/blog/hadoop-ecosystem-and-components

Common Hadoop Distributions

- Open Source
  - Apache
- Commercial
  - Cloudera
  - Hortonworks
  - MapR
  - AWS MapReduce
  - Microsoft Azure HDInsight (Beta)

Setting up Hadoop Development

Hadoop Binaries

Local install
- Linux
- Windows

Cloudera's Demo VM
- Need Virtualization software, i.e. VMware, etc…

Cloud
- AWS
- Microsoft (Beta)
- Others

Data Storage

Local
- File System
- HDFS Pseudo-distributed (single-node)

Cloud
- AWS
- Azure
- Others

MapReduce

Local

Cloud

Other Libraries & Tools

Vendor Tools

Libraries

Comparing: RDBMS vs. Hadoop

	Traditional RDBMS	Hadoop / MapReduce
Data Size	Gigabytes (Terabytes)	Petabytes (Hexabytes)
Access	Interactive and Batch	Batch – NOT Interactive
Updates	Read / Write many times	Write once, Read many times
Structure	Static Schema	Dynamic Schema
Integrity	High (ACID)	Low
Scaling	Nonlinear	Linear
Query Response Time	Can be near immediate	Has latency (due to batch processing)

The Changing Data Management Landscape

MapReduce

- Typical big data problem
  - Iterate over a large number of records
  - Extract something of interest from each **Map**
  - Shuffle and sort intermediate results
  - Aggregate intermediate results **Reduce**
  - Generate final output

Key idea: provide a functional abstraction for these two operations

- Programmers specify two functions:

  map $(k_1, v_1) \rightarrow [<k_2, v_2>]$

  reduce $(k_2, [v_2]) \rightarrow [<k_3, v_3>]$

  - All values with the same key are sent to the same reducer
- The execution framework handles everything else…

Philosophy to Scale for Big Data Processing

Divide Work

Combine Results

Understanding MapReduce

- Map>>
  - (K1, V1) →
    - Info in
    - Input Split
  - list (K2, V2)
    - Key / Value out (intermediate values)
    - One list per local node
    - Can implement local Reducer (or Combiner)
- Shuffle/Sort>>

- Reduce
  - (K2, list(V2)) →
    - Shuffle / Sort phase precedes Reduce phase
    - Combines Map output into a list
  - list (K3, V3)
    - Usually aggregates intermediate values

(input) <k1, v1> → **map** → <k2, v2> → **combine** → <k2, list(V2)> → **reduce** → <k3, v3> *(output)*

WordCount - Mapper

- Reads in input pair <k1,v1>
- Outputs a pair <k2, v2>
  - Let's count number of each word in user queries (or Tweets/Blogs)
  - The input to the mapper will be <queryID, QueryText>:

    <Q1,"The teacher went to the store. The store was closed; the store opens in the morning. The store opens at 9am." >
  - The output would be:

    <The, 1> <teacher, 1> <went, 1> <to, 1> <the, 1> <store,1> <the, 1> <store, 1> <was, 1> <closed, 1> <the, 1> <store,1> <opens, 1> <in, 1> <the, 1> <morning, 1> <the 1> <store, 1> <opens, 1> <at, 1> <9am, 1>

WordCount - Reducer

- Accepts the Mapper output (k2, v2), and aggregates values on the key to generate (k3, v3)
  - For our example, the reducer input would be:

    <The, 1> <teacher, 1> <went, 1> <to, 1> <the, 1> **<store**, 1> <the, 1> <store, 1> <was, 1> <closed, 1> <the, 1> **<store**, 1> <opens,1> <in, 1> <the, 1> <morning, 1> <the 1> **<store**, 1> <opens, 1> <at, 1> <9am, 1>

  - The output would be:

    <The, 6> <teacher, 1> <went, 1> <to, 1> **<store, 4>** <was, 1> <closed, 1> <opens, 2> <in, 1> <morning, 1> <at, 1> <9am, 1>

MapReduce Example - WordCount

- Hadoop MapReduce is an implementation of MapReduce
  - MapReduce is a computing paradigm (Google)
  - Hadoop MapReduce is an open-source software

Big Data Applications

- Link analysis
- Graph data processing
- Data stream mining
- Large-scale machine learning

NoSQL

- Stands for **N**ot **O**nly **SQL**
- Class of non-relational data storage systems
- Usually do not require a fixed table schema nor do they use the concept of joins
- All NoSQL offerings relax one or more of the ACID properties (will talk about the CAP theorem)

Why NoSQL?

- For data storage, an RDBMS cannot be the be-all/end-all
- Just as there are different programming languages, need to have other data storage tools in the toolbox
- A NoSQL solution is more acceptable to a client now than even a year ago
  - Think about proposing a Ruby/Rails or Groovy/Grails solution now versus a couple of years ago

What kinds of NoSQL

- NoSQL solutions fall into two major areas:
  - Key/Value or ‘the big hash table’.
    - Amazon S3 (Dynamo)
    - Voldemort
    - Scalaris
    - Memcached (in-memory key/value store)
    - Redis
  - Schema-less which comes in multiple flavors, column-based, document-based or graph-based.
    - Cassandra (column-based)
    - CouchDB (document-based)
    - MongoDB(document-based)
    - Neo4J (graph-based)
    - HBase (column-based)

Key/Value

Pros:

- very fast
- very scalable
- simple model
- able to distribute horizontally

Cons:

- many data structures (objects) can't be easily modeled as key value pairs

Common Advantages

- Cheap, easy to implement (open source)
- Data are replicated to multiple nodes (therefore identical and fault-tolerant) and can be partitioned
  - Down nodes easily replaced
  - No single point of failure
- Easy to distribute
- Don't require a schema
- Can scale up and down
- Relax the data consistency requirement (CAP)

What am I giving up?

- joins
- group by
- order by
- ACID transactions
- SQL as a sometimes frustrating but still powerful query language
- easy integration with other applications that support SQL