Challenges for Data Driven Systems

Eiko Yoneki

University of Cambridge Computer Laboratory

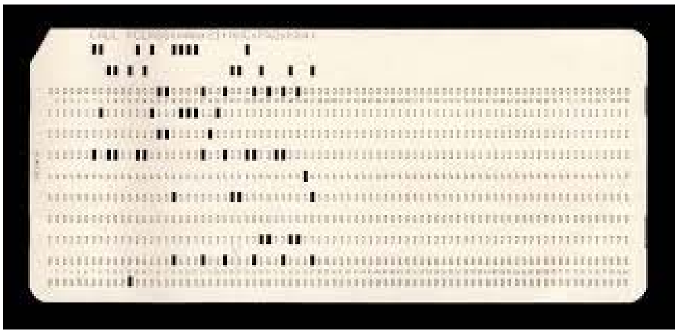
Quick History of Data Management

- 4000 B C Manual recording
- From tablets to papyrus…to paper

A. Payberah'2014

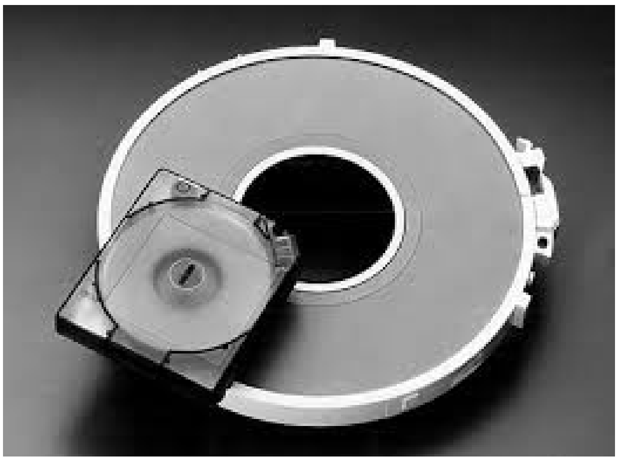
1800's - 1940's

- Punched cards (no fault-tolerance)
- Binary data
- 1911: IBM appeared



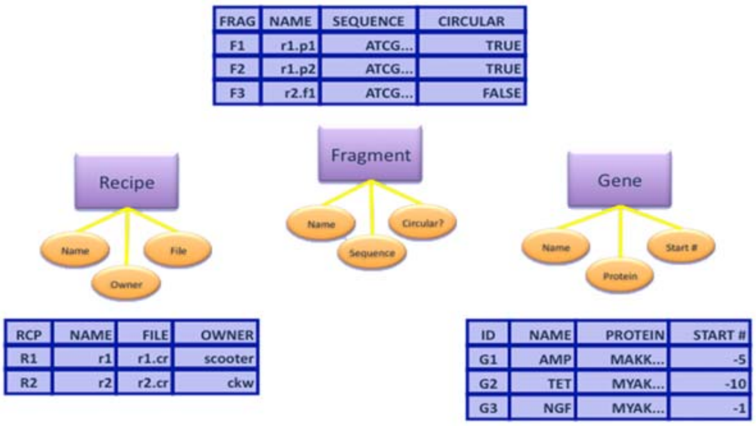
1940's - 1970's

- Magnetic tapes
- Batch transaction processing
- Hierarchical DBMS
- Network DBMS



1980's

- Relational DBMS (tables) and SQL
- ACID (Atomicity Consistency Isolation Durability)
- Client-server computing
- Parallel processing



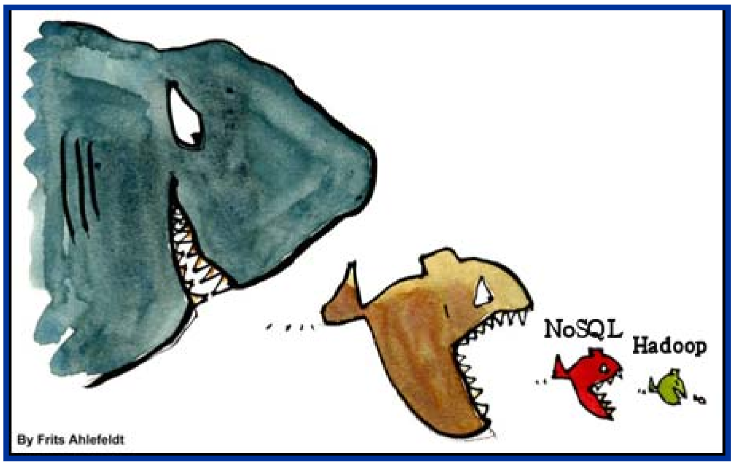
1990's - 2000's

- The Internet...

2010's

- NoSQL: BASE instead of ACID

  Basic **A**vailability, **S**oft-state, **E**ventual consistency
- Big Data is emerging!



A. Payberah’2014

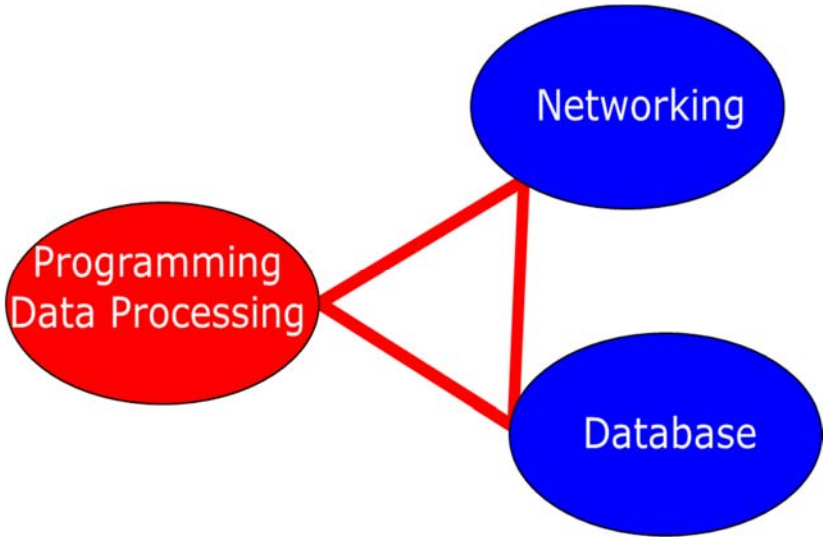
Emergence of Big Data

- Increase of Storage Capacity
- Increase of Processing Capacity
- Availability of Data
- Hardware and software technologies can manage ocean of data

Challenge to process Big Data

- Integration of complex data processing with programming, networking and storage

  → A key vision for future computing



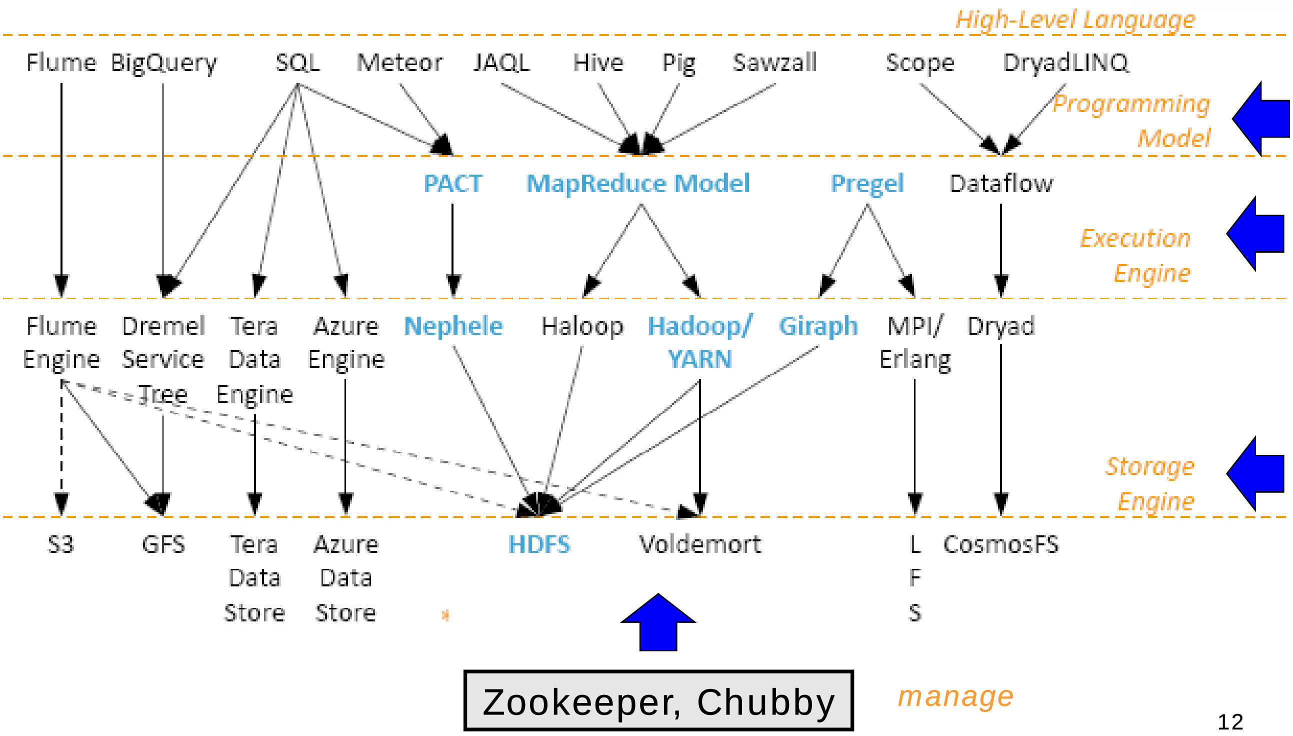
Big Data: Technologies

- Distributed infrastructure
  - Cloud (e.g. Infrastructure as a service)
  cf. Multi-core (parallel computing)
- Storage
  - Distributed storage (e.g. Amazon S3)
- Data model/indexing
  - High-performance schema-free database (e.g. NoSQL DB)
- Programming Model
  - Distributed processing (e.g. MapReduce)
- Operations on big data
  - Analytics

Big Data: Technologies

- Distributed infrastructure
  - Cloud (e.g. Infrastructure as a service)
- Storage
  - Distributed storage (e.g. Amazon S3)
- Data model/indexing
  - High-performance schema-free database (e.g. NoSQL DB)
- Programming Model
  - Distributed processing (e.g. MapReduce)
- Operations on big data
  - Analytics – Realtime Analytics

Distributed Infrastructure



Distributed Infrastructure

- Computing + Storage transparently
  - Cloud computing, Web 2.0
  - Scalability and fault tolerance
- Distributed servers
  - Amazon EC2, Google App Engine, Elastic, Azure
    - System? OS, customisations
    - Sizing? RAM/CPU based on tiered model
    - Storage? Quantity, type
- Distributed storage
  - Amazon S3
  - Hadoop Distributed File System (HDFS)
  - Google File System (GFS), BigTable...

Challenges

- Distribute and shard parts over machines
  - Still fast traversal and read to keep related data together
  - Scale out instead scale up
- Avoid naïve hashing for sharding
  - Do not depend on the number of node
  - But difficult add/remove nodes
  - Trade off – data locality, consistency, availability, read/write/search speed, latency etc.
- Analytics requires both real time and post fact analytics – and incremental operation

Big Data: Technologies

- Distributed infrastructure
  - Cloud (e.g. Infrastructure as a service)
- Storage
  - Distributed storage (e.g. Amazon S3)
- Data model/indexing
  - High-performance schema-free database (e.g. NoSQL DB)
- Programming Model
  - Distributed processing (e.g. MapReduce)
- Operations on big data
  - Analytics – Realtime Analytics

Data Model/Indexing

- Support large data
- Fast and flexible access to data
- Operate on distributed infrastructure

- Is SQL Database sufficient?

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

NoSQL (Schema Free) Database

- NoSQL database
  - Operate on distributed infrastructure
  - Based on key-value pairs (no predefined schema)
  - Fast and flexible
- Pros: Scalable and fast
- Cons: Fewer consistency/concurrency guarantees and weaker queries support
- Implementations
  - MongoDB, CouchDB, Cassandra, Redis, BigTable, Hibase …

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

Big Data: Technologies

- Distributed infrastructure
  - Cloud (e.g. Infrastructure as a service)
- Storage
  - Distributed storage (e.g. Amazon S3)
- Data model/indexing
  - High-performance schema-free database (e.g. NoSQL DB)
- Programming Model
  - Distributed processing (e.g. MapReduce)
  - Stream processing
- Operations on big data
  - Analytics – Realtime Analytics

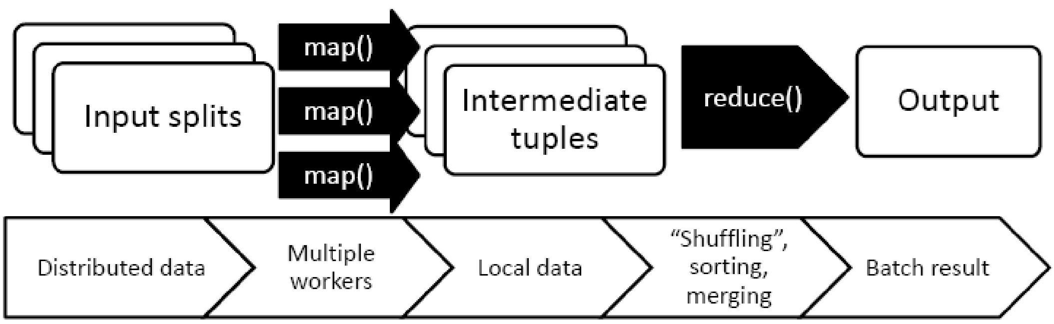
Distributed Processing

- Non standard programming models
  - No traditional parallel programming models (e.g. MPI)
  - e.g. MapReduce
- Data (flow) parallel programming
  - e.g. MapReduce, Dryad/LINQ, NAIAD, Spark

UNIVERSITY OF CAMBRIDGE
Computer Laboratory

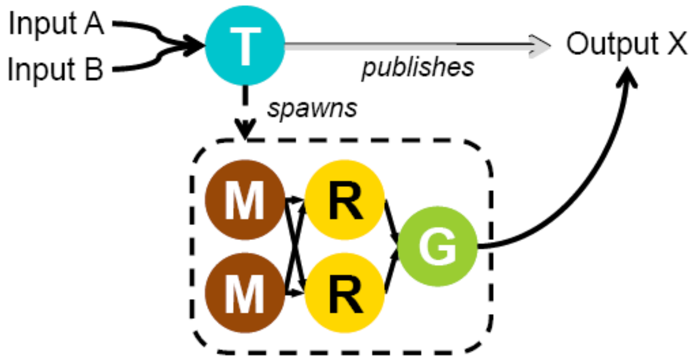
MapReduce

- Target problem needs to be parallelisable
- Split into a set of smaller code (map)
- Next small piece of code executed in parallel
- Results from map operation get synthesised into a result of the original problem (reduce)



CIEL: *Dynamic Task Graph*

- Data-dependent control flow



- CIEL: Execution engine for dynamic task graphs (D. Murray et al. CIEL: a universal execution engine for distributed data-flow computing, NSDI 2011)

Stream Data Processing

- Stream Data Processing
  - Stream: infinite sequence of {tuple, timestamp} pairs
  - Continuous query: result of query in unbounded stream
- Database systems and Data stream systems
  - Database
    - Mostly static data, ad-hoc one-time queries
    - Store and query
  - Data stream
    - Mostly transient data, continuous queries

Real-Time Data

- Departure from traditional static web pages
- New time-sensitive data is generated continuously
- Rich connections between entities

- Challenges:
  - High rate of updates
  - Continuous data mining - Incremental data processing
  - Data consistency

Big Data: Technologies

- Distributed infrastructure
  - Cloud (e.g. Infrastructure as a service)
- Storage
  - Distributed storage (e.g. Amazon S3)
- Data model/indexing
  - High-performance schema-free database (e.g. NoSQL DB)
- Programming Model
  - Distributed processing (e.g. MapReduce)
- Operations on big data
  - Analytics

Techniques for Analysis

- Applying these techniques: larger and more diverse datasets can be used to generate more numerous and insightful results than smaller, less diverse ones

- Classification
- Cluster analysis
- Crowd sourcing
- Data fusion/integration
- Data mining
- Ensemble learning
- Genetic algorithms
- Machine learning
- NLP
- Neural networks
- Network analysis
- Optimisation
- Pattern recognition
- Predictive modelling
- Regression
- Sentiment analysis
- Signal processing
- Spatial analysis
- Statistics
- Supervised learning
- Simulation
- Time series analysis
- Unsupervised learning
- Visualisation

Typical Operation with Big Data

- Smart sampling of data
  - Reducing data with maintaining statistical properties
- Find similar items
  - Efficient multidimensional indexing
- Incremental updating of models
- Distributed linear algebra → dealing with large sparse matrices
- Plus usual data mining, machine learning and statistics
  - Supervised (e.g. classification, regression)
  - Non-supervised (e.g. clustering..)

Do we need new Algorithms?

- Can’t always store all data
  - Online/streaming algorithms
- Memory vs. disk becomes critical
  - Algorithms with limited passes
- N^2 is impossible
  - Approximate algorithms

Easy Cases

- Sorting
  - Google 1 trillion items (1PB) sorted in 6 Hours
- Searching
  - Hashing and distributed search

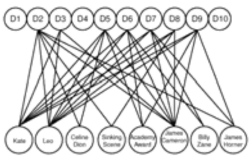
→ Random split of data to feed M/R operation

- BUT Not all algorithms are parallelisable

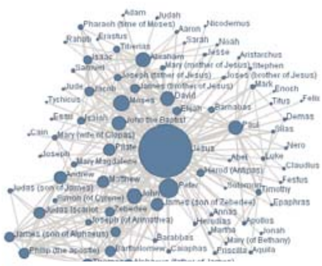
More Complex Case: Stream Data

- Have we seen x before?
- Rolling average of previous K items
- Hot list–most frequent items seen so far
  - Probability start tracking new item
- Querying data streams
  - Continuous Query

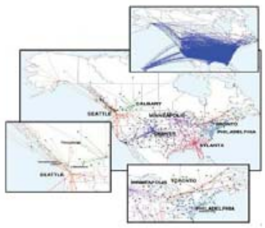
Big Graph Data



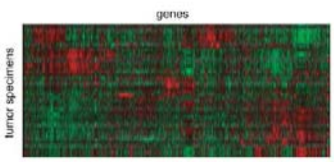
Bipartite graph of appearing phrases in documents



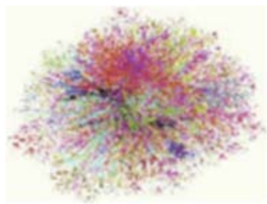
Social Networks



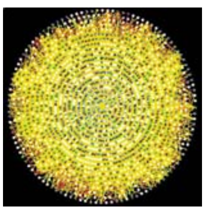
Airline Graph



Gene expression data



Internet Map [lumeta.com]



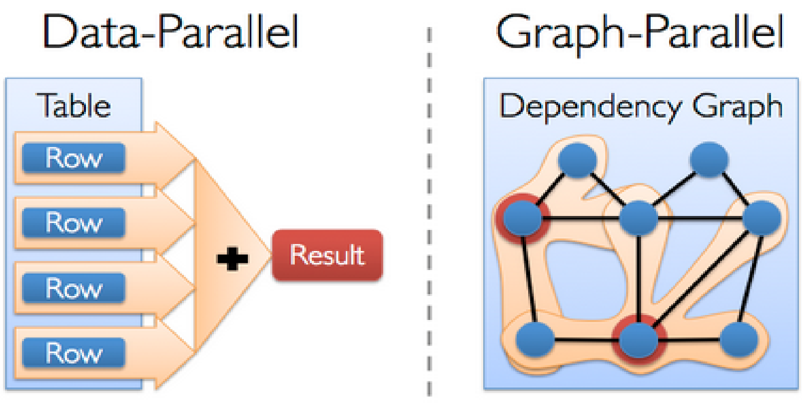
Protein Interactions [genomebiology.com]

How to Process Big Graph Data?

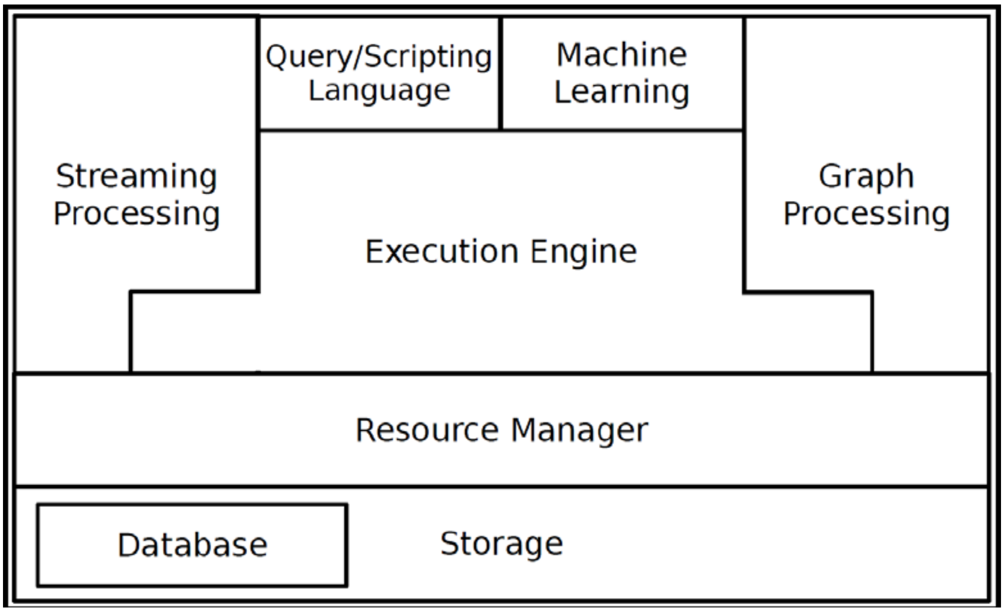
- Data-Parallel (MapReduce, DryadLINQ)
  - Partitioned across several machines and replicated
  - No efficient random access to data
  - Graph algorithms are not fully parallelisable
- Parallel DB
  - Tabular format providing ACID properties
  - Allow data to be partitioned and processed in parallel
  - Graph does not map well to tabular format
- Moden NoSQL
  - Allow flexible structure (e.g. graph)
  - Trinity, Neo4J, HyperGraphDB
  - In-memory graph store for improving latency (e.g. Redis, Scalable Hyperlink Store (SHS))

Big Graph Data Processing

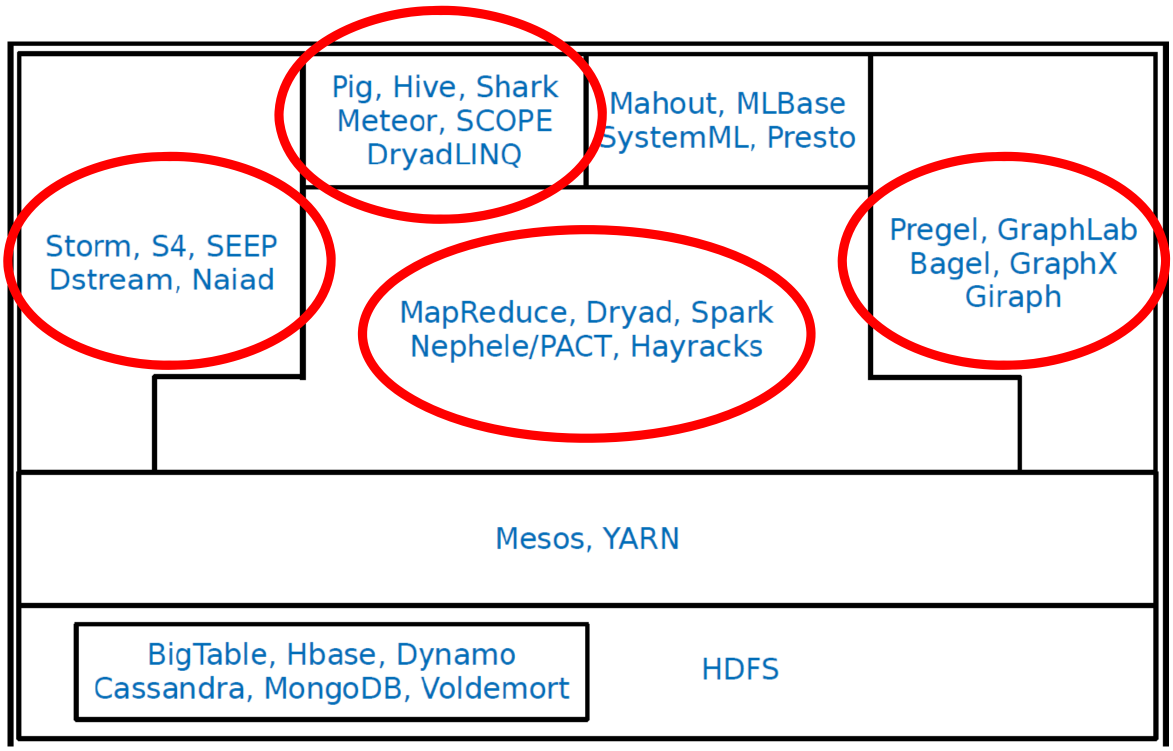
- MapReduce is ill-suited for graph processing
  - Many iterations are needed
  - Intermediate results at every iteration harm performance
- Graph specific data parallel
  - Vertex-based iterative computation model
  - Iterative algorithms common in ML and graph analysis



Big Data Analytics Stack



Big Data Analytics Stack



Topic Areas

Session 1: Introduction

Session 2: Programming in Data Centric Environment

Session 3: Processing Models of Large-Scale Graph Data

Session 4: Map/Reduce Hands-on Tutorial with EC2

Session 5: Optimisation in Graph Data Processing
+ Guest lecture

Session 6: Stream Data Processing + Guest lecture

Session 7: Scheduling Irregular Tasks

Session 8: Project study presentation

Summary

- R212 course web page:

www.cl.cam.ac.uk/~ey204/teaching/ACS/R212_2014_2015

- Enjoy the course!